Icon 26 is readily expanded by adding further participants $A = (\pi_A^1, \pi_A^0)$; say A_1 A_2 ..... each initially attached to a brain, say α_1, α_2 ..... The expanded icon represents a many person application of CASTE. Heuristic B is still dominant. The support $< S_B^1, S_B^0 >$ is executed at the interface (to which β is reduced). But, using a design based upon earlier work with small groups (Pask and Von Foerster 1961, Lewis and Pask 1966 and Pask and Lewis 1968) it is possible to distribute $< T^1, T^0 >$ for execution in some or all of the L Processors α_1, α_2 ..... This, at any rate, is a workable realisation and it is described in the next volume.

As a preview, not only is $< T^1, T^0 >$ distributed under execution but so are the P Individuals (A_1, A_2 ....). The student-to-student interactions that are catalysed in realising $< T^1, T^0 >$ ensure that A_i is executed to some extent in α_i and A_j in α_i. The main technical expedient is separate storage of the marker predicate assignments for each currently operating P Individual and replication of the support $< S_B^1, S_B^0 >$. Incidentally, the number of P Individuals is not necessarily the same as the number of students entering the system; coalitions are formed and fresh P Individuals may, in principle, be constructed as deviants or by co-operative combination.

In contrast to the idea of a "teaching machine", which was bandied around and mildly derrogated in Chapters 1 and 2, this arrangement is a "vehicle for driving through knowledge". That is, a student sees the entailment structure as a terrain through which he may progress, more or less freely, by learning and understanding. But he is subject to navigation rules imposed by the CET heuristic of (S_B^1 S_B^0) and boundary conditions that are imposed, through the macro measures discussed earlier in this chapter, perhaps augmented by FRIM indices. Further, the student expresses his desired direction and rate of motion in terms of macro variables. CASTE, regarded as a physical processor, provides an M Individual which the student (qua P Individual) inhabits and uses as a vehicle to drive through the terrain of the entailment structure. Each student may be seen (and may see himself) as occupying a separate vehicle. But students converse with one another, as well as with the support $< S_B^1, S_B^0 >$. In fact, student transactions carry the burden of teaching. Students are permitted to converse, if they occupy the same neighbourhood in the terrain and if they have learning strategies that are compatible.

General applications of SST systems are reviewed in Lewis and Pask (1964) and in Pask (1974); a summary account of the gross findings is given in Pask (1966). Detailed descriptions appear in Pask (1958, 1960, 1961, 1963, 1965, 1969). Most of the experimental work referred to the acquisition of coding skills in which a subject is presented with a complex stimulus (a configuration of several illuminated signal lamps, for example) and is able to achieve a correct result if he makes an appropriate complex response (pressing a configuration of several response buttons, for example) within a fixed interval of Δt. The response is appropriate if and only if the buttons pressed satisfy a coding rule that relates the display (of lamps) to the set of response buttons; Δt and the intertrial interval are chosen so as to render the task learnable but impossible for any novice. In our more sophisticated experiments, we regulated Δt to fit the individual subjects and took care, when analysing the data, that such individual adjustments did not pervert the data, at any rate in a direction that favoured the hypothesis being tested.

Most of the experiments were controlled by a special purpose computer on which any kind of SST or adaptive system could be simulated; the display arrangements included means for delivering either simple or differential 'knowledge of results' for optionally cueing in respect of each or all response components and for changing and alternating the current rule. The work is chiefly reported in Pask and Lewis (1967, 1968).

Salient conditions and results were as follows (each experimental subgroup has between 10 and 20 subjects).

(1) Using a single rule, and simplifying the problem (of realising the rule for a randomly selected sequence of lamp configurations within Δt) by varying the lamp number and by an auxiliary cueing adjustment, it was possible to confirm the result cited in the text. Learning rate is maximised by setting ξ so that problems are maximally difficult whilst remaining intelligible; for values of ξ spaced 1/10 ρ apart, the predicated trend is significant at the 0.1% level. The auxiliary cueing is introduced to avoid a pathological inversion, noted by Van Der Veldt and cited in Woodworth

(1950), due to the fact that subjects build up a mental scaffolding when dealing with an increasing number of response components and this structure is prone to collapse. (Thus, for all subjects, 4 lamp problems are more difficult than 3 or 2 lamp problems when initially attacked; subsequently, for some subjects, 3 or 2 lamp problems may not be less difficult than 4 lamp problems unless auxiliary cueing is employed.)

(2) For two or more randomly alternated rules and the requirement of responding correctly at any trial (given the name of the rule) the task is much harder to perform. If separate subcontrollers are used to adjust the problem difficulty for each subtask, a similar ξ trend is obtained but many subjects have a tendency to 'play with' the system; we called this phenomenon 'Participant Interaction' and noted that it depends upon the subject's learning the regulator characteristics and making contrived rather than veridicial responses in order to induce desired regulator behaviours. It is also evident, even at the miniscule grain of response latency groupings, that subjects adopt very definite problem solving methods.

(3) The following conditions (a, b, c) were employed to examine the relative efficacity of different teaching strategies; in condition (a) all of the strategies counter or accommodate the influence of Participant Interaction. In conditions (b) and (c) below participant interactions are legalised as meta statements about problem solving.

(a) Adaptive alternation of rules intended to minimise the interference between acquisition of the subskills needed to cope with each rule.

(b) Free learning. The subject has free choice of a current rule over a block of trials, contingent upon the requirement that he ultimately gains proficiency in respect of all rules when they are randomly and equiprobably alternated at successive trials.

(c) A compromise strategy which uses the 'metainformation' from condition (b) and the adaptive strategy of condition (a) depending upon which is the most effective, so far as the individual subject is concerned. This arrangement is called an adaptive 'metasystem'.

In all conditions, the terminal criterion is to achieve proficiency at the maximum level of difficulty (for each rule) if all rules are randomly and equiprobably alternated over a sequence of trials.

(4) Condition (c) yields, as predicted, higher learning rates than conditions (a) or (b) and, in this respect, condition (a) is superior (for most subjects) to condition (b). The main results are as indicated below (10 subjects in each subgroup; confirmed for larger subgroups in the earlier study).

Trials to criterion scores

	Mean	S.D.
Condition (a)	76.2	13.14
Condition (b)	116.4	20.81
Condition (c)	60.	9.5

Mean (a) > mean (c), just significant, at 5% level; mean (b) > mean (a), significant at 0.1% level; mean (b) > mean (c), significant at 0.1% level; trend "condition (c) better than condition (a) better than condition (b)" significant at 0.1% level.

A specific index of interference between the subskills is minimised by condition (c); here the (c) to (a) difference is significant at the 1% level.

(5) Very similar comments apply to rule composition in series as well as rule alternation (so that the input/output mapping is determined by an iterated transformation). The pertinent data are only published in a Technical Report (Pask, Lewis, Feldman, Robinson, Watts, 1966).

In typical experiments the task involved 2 or 3 hours of learning, rest intervals apart. Since the experimental regulators were set up on a special purpose machine it was possible to obtain very detailed records and consequently it was worthwhile modelling the system at a structural rather than statistical level. Mr. Feldman and Mr. Mallen wrote several computer programs to exhibit the activity of these models. The following features are distinctive and of some interest.

(a) The programs imaged the entire man-machine system; that is, part of the model (by far the largest part) represented the subject; a smaller (interacting) part represented the regulator. Amongst other things, parameters of the programmed regulator were changed and (using simulation data obtained from comparing differently parameterised runs) it was possible to modify and

redesign the real life regulators. By observing a real life experiment with the redesigned regulator and iterating the cycle, it was possible to improve the student model.

(b) Subject models consisted in processes for building up structures including strings and complexes of basic operations; for substituting, replacing and copying elements of these structures, and for generating complex responses. A further compartment of the subject model acted as an executive for governing the effort allocated to these activities in terms of a computed variable representing subjective uncertainty.

(c) Using such joint or systemic models it was possible to match real and simulation data at the grain of response component latencies upwards and to account for the salient instabilities due to interference, the Van Der Veldt inversion effort, and the genesis of Participant Interaction. The simulation and iteration studies are described in two technical reports (Pask, Lewis, Feldman 1966 and Pask, Lewis, Mallen, Robinson 1967: briefly in Pask 1965, Pask and Scott 1971, with greater generality in Pask 1974).

Appendix B

The cooperative externalisation technique (CET) was used in connection with a code learning task (Pask and Scott 1971) formally isomorphic to the task described in Appendix A. The display and response facilities are shown in Diagram B1. Signal lamps designate the values of attributes A, B, C, D (rather than stimulus elements) but the response components retain the (solution indicating) status of the previous study.

A systematic meta-dialogue is established between the subject and the regulator, as a means for externalising the way in which the subject partitions the relation characterising the entire task (the Code Rule). This dialogue has two constituents. On the one

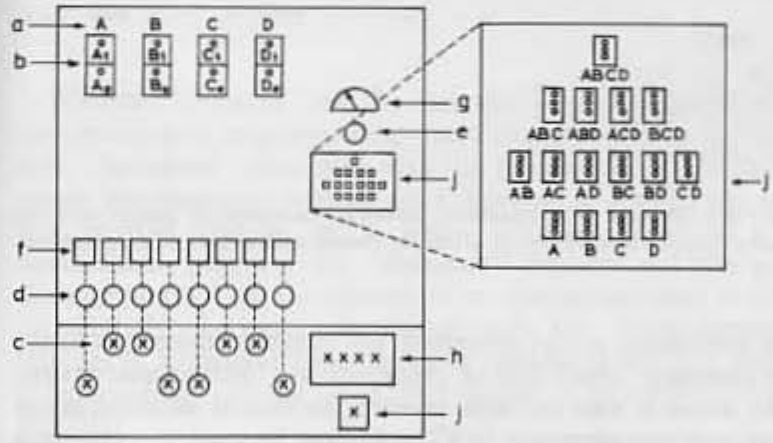


Diagram B.1. Display and response board. a = Binary attribute names, A,B,C,D; one for each box. b = Stimulus lamps, A_1, A_0; B_1, B_0; C_1, C_0; D_1, D_0; in boxes. The lamp in the upper box indicates the value 1 of an attribute, and the lamp in the lower box, the value 0. Stimuli, x, are configurations of illuminated lamps (see text). c = Eight response keys, $y_1 \ldots y_8$. d = External register, retaining key-pressed information until the end of any trial. e = Partial knowledge of results lamp (complete response is or is not correct). f = Complete knowledge of results lamps indicating correct value for each component of the response. g = Score meter. h = Attribute selection buttons. i = Submit button. j = Last value of score, display. Three lamps for each subset: blue (upper), $\rho > 0.75$; green (middle), $0.75 \geqslant \rho \geqslant 0.5$; and red (lower), $0.5 > \rho$.

hand, the subject is able to direct his attention to subrelations associated with subsets of the attributes A, B, ... A, B: A, C: A, D: B, C: . . . . ABC, ABD, ... A, B, C, D. A subset is selected by pressing a number of attribute selection buttons and, during the subsequent block of trials, problems within this partition are rehearsed exclusively and under the control of an adaptive regulator. As before, there is a general caveat that ultimately the subject must tackle problems circumscribed by ABCD (equivalent to the "4 lamp problems" of the previous study). On the other hand, the regulator furnishes a performance description by displaying, in the format of a task description, the subject's previous level of proficiency in respect of each subset selected. The descriptive format is shown in Diagram B2; the code rule is a mapping from the "space" with A, B, C, D, acting as its coordinates, to the unordered (but labelled) set of response buttons.

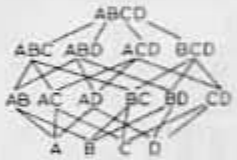


Diagram B.2. Subtask or subproblem hierarchy obtained by partial ordering of classes. Rule relates values of attribute classes and subset of response set.

The conditions so far described are roughly equivalent to the "Free Learning" condition of Appendix A. Under these circumstances, subjects who are able to learn the task at all adopt rather definite learning strategies (CET exhibited through the attention directing interaction). One strategy type is called "stringing" (Diagram B3(I)) in which a typical exploration sequence is

$$A \rightarrow B \rightarrow AB \rightarrow C \rightarrow ABC \rightarrow D \rightarrow ABCD$$

Another typical strategy is called "grouping" (Diagram B3(II)): it is characterised by the assembly and integration of fairly complex subtasks; as indicated by the exploration paradigm.

$$A \rightarrow B \rightarrow AB \rightarrow C \rightarrow D \rightarrow CD \rightleftarrows AB \rightarrow ABCD$$

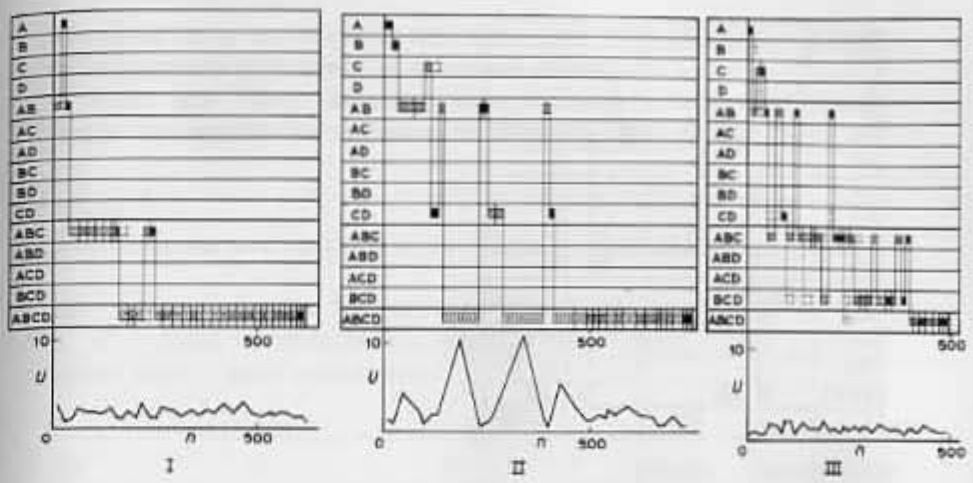


Diagram B.3. Typical free-learning curves from expt. 2. Upper part of figure: problem class selections along the vertical coordinate; n (number of trials), along the horizontal. Proficiency, ρ, is shown by shading for each block of trials: ■ $\rho \geqslant 0.75$; ▤ $0.75 > \rho > 0.25$; ▭ $0.25 \geqslant \rho$.

Whereas "stringing" (at the strategic level) correlates with the construction of response component strings at the level of latency data, "grouping" correlates with the production of "clumps", or nearly simultaneous component responses, as though the subject was playing chords on a piano. Further, there is a close resemblance between the "stringing" strategy and the serialist strategy mooted (in the context of an intellectual task) in Chapter 3: similarly, between "grouping" and the holist strategy of Chapter 3. Though intermediaries or variants exist (they may exist in intellectual learning, also) they appear to be hybrids (Diagram B3 (III)).

Success in learning depends upon a match between the free learning strategy adopted and the subject's competence to execute the strategy he has chosen. Since response component latencies are correlated with strategic selection, it is possible to estimate competence quite early in the learning process and to impose or recommend a strategy matched to the subject's competence. The gross data are mostly concerned with experiments in which "Free Learning" is compared with learning regulated by an advice-giving "adaptive metasystem" (detailed in Pask 1969, 1970) which secures matching. Learning records obtained using the adaptive

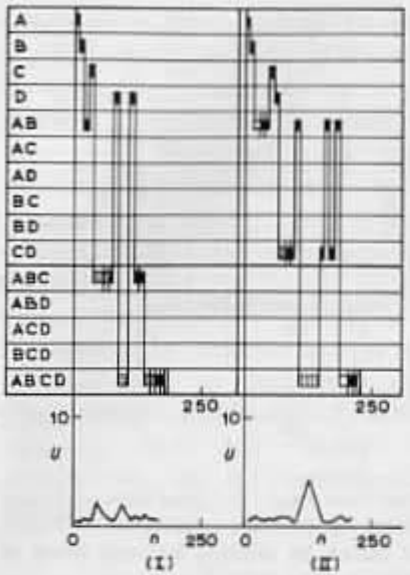


Diagram B.4. Typical learning curves for metasystem subjects.

metasystem are shown in Diagram B4 (I) (a subject where the system assigns a stringing strategy) and in Diagram B4 (II) (where the adaptive metasystem assigns a grouping strategy). A subsidiary enquiry was concerned with the efficacity of a simple adaptive regulator, based on one of the strategies (according to one criterion, the logically provident strategy).

Since individual variations are quite informative, the data from the main study is presented in Table 1, in terms of T, the number of trial blocks needed to achieve a criterion. The length of a block is variable due to the sequential method employed to estimate p (see Pask and Scott, 1971; Scott 1970) but a lower value of T is a veridicial indication that the rate of learning is enhanced. The experimental design involved two main sessions during which subjects learned different rules (1 and 2) in an attempt to ascertain whether the first session strategy is carried over into the subsequent session. The answer to this question is unequivocal: if a strategy is determined explicitly for Rule 1 it is employed for Rule 2. The entire operation was preceded by a further session, used to ensure that subjects were familiar with the equipment and to adaptively determine the individually appropriate value of maxium (correct) response interval, Δt. Various strategic protocols were

obtained from the subjects and these tally with the CET records; other sessions were used for retention testing.

From the "Free Learning" data in Table 1 it is evident that there is considerable variation in learning rate. It also turns out

Table 1
Free learning and Learning in an Adaptive Metasystem

Δt = interval allowed for response. T_1 = trials to criterion for task 1. T_2 = trials to criterion for task 2. $T^*(\%)$ = ratio $T_1/T_2 \times 100\%$. Mean U(n) = mean uncertainty.

Δt	T_1	T_2	$T^*(\%)$	Mean U(n)
Free learning				
3.23	329	313	64.1	0.84
3.02	236	558	42.3	0.77
2.55	77	249	30.9	0.73
3.23	181	623	29.0	1.27
3.02	77	488	15.4	0.71
3.13	147	380	38.7	0.70
3.02	121	300	24.7	0.95
3.42	302	748	40.4	1.9
3.5	183	610	30.0	1.64
3.23	121	475	25.5	1.0
Adaptive metasystem				
3.32	297	220	135.0	0.44
3.13	184	286	64.3	0.57
3.23	184	276	66.7	0.39
3.02	207	180	115.0	0.52
2.55	104	148	70.3	0.50
3.13	133	209	64.6	0.44
3.05	196	229	83.5	0.53
3.07	174	211	82.5	0.45
3.12	203	208	97.5	0.46
3.00	167	232	72.5	0.52

that if a strategy is matched to the subject's competence either strategy is just about as effective. The main comparison is effected between the Rule 2 subjects in the "Free Learning" and the "adaptive metasystem" subgroups; using T_2. The difference is highly significant (0.1% but the rank scores never overlap). The same significance is attached to the check ratio score $T^* = T_1/T_2 \times 100\%$.

The lower lines of Diagram B3 and Diagram B4 represent an index of uncertainty calculated from the number of guessing trials required to make a completely correct response to each input (ρ, in contrast, is a correct response index). It is illuminating to plot the uncertainty (U) values, in aggregate over the "Free Learning" and "adaptive metasystem" subgroups, but with reference to each subtask. These graphs, shown in Diagram B5, reveal the dominant effects of the adaptive metasystem; this method of regulation reduces peaks of gross uncertainty and prevents premature rehearsal of the complete task (ABCD) which engenders uncertainty if the prerequisite subtasks are not already familiar and proficiently performed.

Table 2 shows the data from the experiments using an adaptive regulator designed on the basis of *one* logically provident type of learning strategy. As in Appendix A, the adaptively regulated subjects fair better than most free learning subjects; the difference

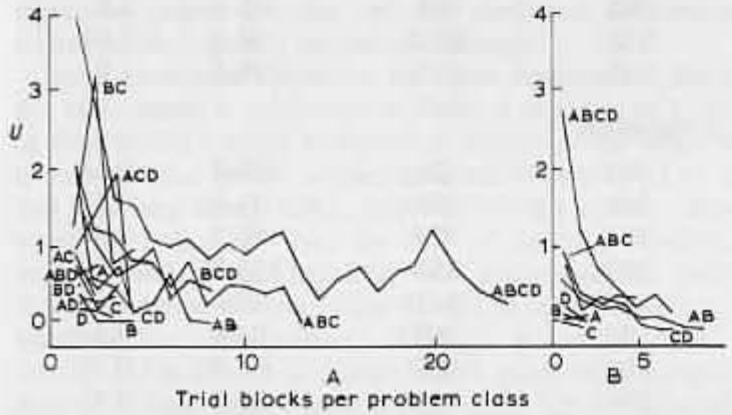


Diagram B.5. Plot of U (behavioural uncertainty), averaged over A, free-learning subjects, B, metasystem subjects with respect to separate problem classes. Plotted against number of trial blocks spent dealing with each class.

Table 2
Adaptively Regulated Learning

T_2	Δt
424†	3.02
510	3.42
400	2.75
368†	3.10
310†	3.23
256†	3.00
299†	3.32
621	2.85
398†	3.05
444	3.30

†Matched.

is significant at the 0.1% level. However, there is a great deal of variation and the order of superiority may be reversed for certain free learning subjects who have already "learned to learn". It is also intuitively clear that the adaptive subjects who are the best performers have an ingrained disposition to adopt the *one* learning strategy built into the adaptive machine. In Table 2 these subjects are distinguished by a special mark.

Appendix C

Typical Teachback Protocols for Task II

KEY

n^{-} ... $^{-n}$	Information derived from frame *n*
———	Inventions
———	Irrelevant information
.....	Redundant information
———	Statement referring to information to be presented or previously mentioned
~~	Information inferred from the programme by the subject.
———	A false statement.
+	Cross reference of a false statement with a correction.
[ ]	Repetitions.
( )	Other comments, interjections, etc.

(i) *Holist with holist programme*

(1) I'm going to tell you about a ^{1}funny Martian animal which has been recently discovered and classified by scientists conducting surveys^{-1}. They are funny slug-like things with ^{1}various protruberances, some of which, differ amongst the different types. Zoologists use these differences to draw up a classification^{-1} ^{2}There are other ways of telling them apart$^{-2\ 7}$ The main alternative is a Russian scheme which uses masticator differences where the standard scheme talks about sprongs$^{-7\ 8}$ which are a kind of horny spike, used for defence against predators, notably the Owzard or night vulture which is also Martian^{-8}

(2) ^{3}These animals are called Gandlemullers,$^{-1\ 3}$ because they churn about in the swamps near the equator and Gandle is Martian for swampmud, hence smampmudmiller—muller is German for Miller^{-3}. [These things churn through the mud], ^{8}eating it by some curious process which means they eat and excrete at the same time^{-8}—like a flow through bakery.

(3) [Anyway, they sit about eating this stuff] and all ^{11}the time they wave tendrils in the air called vibratory sensors for detecting Owzard's wing beats^{-18}. When attacked, they do various things depending on their defence equipment. For a start ^{18}some of them retract their sensors. Some don't because they are fixed$^{-14\ 15}$ Some of them lie all flat and squashed because they have no spikes for stabbing.$^{-15}$ Others have these sprong things and do various gymnastics. If ^{11}they have a front and back spike they jackknife.$^{-11}$ If ^{13}they have only a front one, they rear up on a thing at the back called a ventral skirt^{-13}. If ^{14}their spike is at the back, they do it the other way on a thing called a foreplank^{-14} which, by the way, ^{19}is collapsed by the ones that have no sprongs so they can get flat^{-19}.

(4) Anyway, if we come to the classification proper you'll see how they differ in the various ways.

(5) (Now) [1] [2]There are 13 different subspecies [3] [4]divided into 3 main types. There are [12] [5]4 Gandlers, [6] [11] [16] [7]4 Plongers [8] [15] and [23] [9]5 Gandleplongers [10], [22].

(6) You [17] [18]get these with various numbers of bodies—one, two or three and a subspecies has a prefix M(mono), B(bi), T (tri) to say which is which [19].

(7) There [11] [20]are 2 M-Gandlers, 2B-Gandlers. [21]. Similarly there [18] [22]are 2 M-Plongers and 2 B-Plongers [23].

(8) Now [1] [24]in Gandleplongers you get the T-types [25] Oh, [12] [26]there are 2 sorts of Gandleplonger called [13] [27]α or Ω. [14] [28]depending on their spong. You see they have only one sprong [29], α is the one with it at the front [30] and Ω is at the rear [31]. They're the ones that pivot at the front or back depending, as I said before. Anyway, [23] [32]there are 2 α's an M and a T and there are 3 Ω's, one B and two T's [33]. I'll say how you tell them apart in a minute, but first I must say about sprongs.

(9) [19] [34]Gandlers are the ones with no sprongs [35] [11] [36]Plongers are the ones with 2 sprongs [37] and [12] [38][Gandleplongers have one at the front or back [39], as I just said[13] [40]α and [14] [41]Ω [42], [14]].

(10) Now I'll do Gandlers. You [15] [43]have to tell the two M's from the B's. This is by cranial mound which is a bump on the head. The MI and BI have a single bump. The MII and BII have a double bump which [16] [44]shows they are very clever. So I means single, II means double [45], [16].

(11) Now Plongers. [16] [46]There's an Ma and Mb and a Ba and Bb. [47]. Here we use two different tests. One of them we use again in a minute.

(12) Now [18] [48]the M-Plongers are distinguished by the vibratory sensor and you find that the M-Plonger-a has a fixed sensor[49] (no sorry, retractible[49]), whereas the Mb is fixed [50]. Turn now to the B's[23] [51]. Here we find that one of them has a hairy dorsal mound. Hairiness is a good thing because it's an insulation against heat loss in digestion.

(13) The one with the hairy mound is B-Plonger-b [52]. Anyway that's the Plongers, [the prickers with 2 sprongs.] The last ones are the Gandleplongers.

(14) [As I said there are 2 sorts of α and Ω's. [53] [54]The α's are Mα and Tα [55].] That's all we need to know number of bodies and sprong type. Now [23] [56]we have one BΩ on its own. And two TΩ's [57]. There [21] [58]are Ωa and Ωb there we find a difference we used before. It's vibratory sensor and again its the a which is retractible. The b is fixed [59] (That should do).

(ii) *Serialist with serialist programme*

(1) [1] [60]Zoologists have classified the Gandlemuller on the basis of physical characteristics [61].

(2) [2] [62]The three main types are Gandlers, Plongers and Gandleplongers [63].

[2] [64]Gandlers have no sprongs. Plongers have two sprongs. Gandleplongers have one sprong [65].

(3) [4] [66]There are four subdivisions of Gandler; MI, MII, BI and BII [67] [5] [68]The M's have one body, the B's have two bodies. The MI and BI have a single cranial mound. The MII and BII have a double cranial mound. [69].

(4) [6] [70]There is an M-Plonger-a and b and B-Plonger-a and b [71] [7] [72]M's with one body and are distinguished by the type of vibratory sensor, which for M-Plonger-a is retractible. The b is fixed.

(5) The two B-Plongers have two bodies and are distinguished by type of dorsal mound. The B-Plonger-a has a smooth dorsal mound, the "b" has a hairy one [73].

(6) (Let's see,) [8] [74]there are 2 Gandleplonger-α's and 3 Ω's [75].

(7) [9] [76]The α's are the Mα and the Tα with one and three bodies respectively [77].

(8) [10]The Ω's have one B and two T types, TΩa and TΩb [78].

(9) (Oh) [9] [79]α's have a foresprong, Ω's have a tailsprong [80] and [11] [81]the T-Gandleplonger Ωa has a retractible vibratory sensor whereas the b is fixed [82].

(iii) *Holist with serialist programme*

(1) Now, [2] [83]there are 3 main types of [1] [84]Gandlemuller [85]. [4] [86]There are four called Gandlers[4] [87], [6] [88]four called Plonger [89] and [8] [90]5 called Gandleplonger[8] [91]. That's 13 altogether.

(2) The group of 5 come in two kinds α or Ω depending on a physical characteristic [92] (Oh yes) [3] [93]Zoologists have made a classification based on physical [94] characteristics.

(3) [5] [95]Some of them have spiky things, some don't [96].

(4) Some have 1 body, some have 2 or 3.

(5) You also have various hairy or smooth, single or double mounds.

(6) There's something else. Retractible vibratory sensors, or fixed. That's used twice.

PAUSE

(7) (Let's see). [4] [97]There are 4 Gandlers, M and B [98] [6] [99]4 Plongers M and B [100] and [8] [101]5 Gandleplongers [102] [9] [103]M & T of one sort [104] and [10] [105]M and B of other [106] (No, B and T[+]) You [4] [107]tell the Gandlers apart by a suffix I or II [108] telling you whether [5] [109]they have this mound single or double [110].

(8) You tell [6], [7] [111]Plongers apart by sensors, a fixed b retractible [112], [7] (It might be the wrong way there) (I think) [7] [113]you also get the mound in here. Hairy is b [114].

(9) The [9] [115]Gandleplongers have a thing which is fore or aft. You put α or Ω after the name [116].

(I've missed something, but I can't really remember all the bits. I got quite a lot though).

(iv) *Serialist with holist programme*

(1) In [1] ¬surveys of Mars, Zoologists have classified Gandlemullers into 13 subspecies depending on physical characteristics ¬[1].

(2) [2] ¬There are also differences in behavioural habits ¬[2].

(3) [3] ¬Gandle is mud. Muller is miller ¬[3].

(4) They [4] ¬have various physical things that are used . . . There are sprongs, foreplank and a ventral thing. Masticators, an aperture and a dorsal mound ¬[4].

PAUSE

(5) [5] ¬There are 3 main subspecies: M [6] ¬means mono, B means Bi and T means tri, for number of bodies ¬[5]. The names are Gandler, Plonger and Gandleplonger ¬[6]

[7] ¬They live in caves ¬[7]

(6) [10] ¬Gandlers have no sprongs ¬[10] [11] ¬Plongers have 2 ¬[11] and [12] ¬Gandleplongers have 3 ¬[12].

(7) [[8] ¬They have masticators] for eating mud ¬[8] [9] ¬which takes 2½ minutes and is stored ¬[9].

(8) [10] ¬When Owzards, the Martian night vulture, attacks Gandlemullers, he lies flat excreting all mud ¬[10]. (No, that's just Gandlers)+ because [11] ¬Plongers prick or stab and jacknife ¬[11].

(9) Some have [12] ¬spoon-shaped masticators which are hairy or smooth ¬[12].

(10) You [13] ¬get M-Gandlers and B-Gandlers with suffix I and II meaning single or double cranial mound ¬[13].

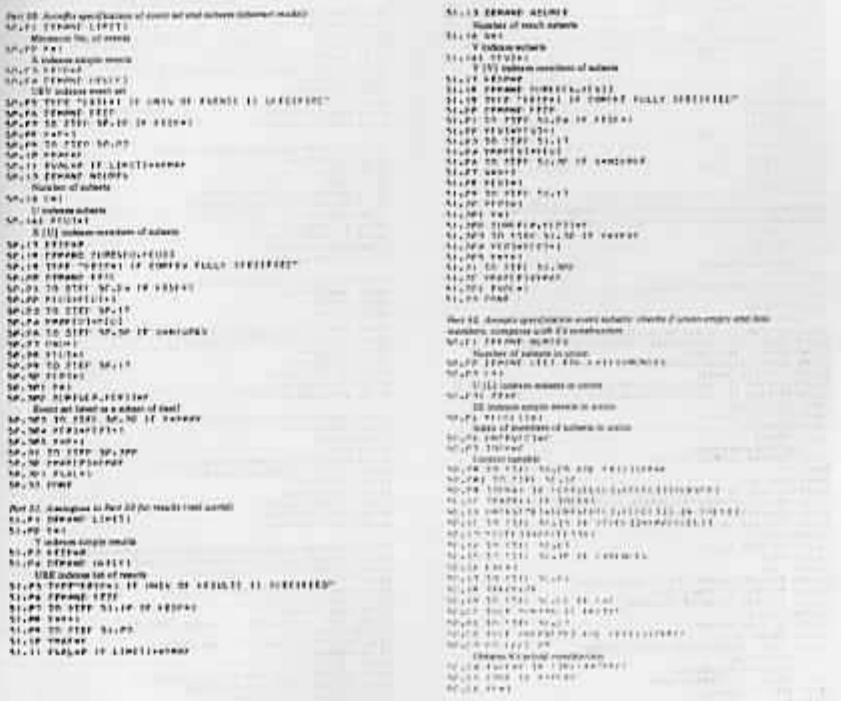
(11) You get [14] ¬α or Ω meaning retractible or fixed sensor ¬[14]. (I can't remember any more with much clarity). I know some of them [24] ¬swim in various ways ¬[24]. And its a [22] ¬B-Plonger-b that has a hairy something ¬[22]. And [24] ¬ there are TΩa's and b's depending on the number of spikes ¬[24].

Appendix D

The CASTE operating programme is listed in TELCOMP II; consequently it is addressed to a terminal user. The user in question is not the student but may be an experimenter who inputs data to the teletypewriter (for each *Demand*), the data being copied from registers in the interface processor. Similarly teletypewriter outputs are input to the interface processor and act upon the display (for example, setting up markers for Aim and for Workset). Since the experimenter does nothing (in this capacity) apart from copying data, his duties as scribe are normally delegated to the processor itself; its registers are loaded from, and unloaded into, the program by a direct connection from the modem. For this purpose the program is augmented by statements which make it "Type" (virtually, that is) certain expressions only interpreted by the interface processor; these statements are not shown in the listing. Similar comments apply to the task structure program in Appendix E which is also listed in teletypewriter terminal form.

Appendix E. Task structure routines for elementary probability theory

The listed program is set out in the same form as the Appendix D listing.



1. P Individuals are characterised in terms of the theory of self reproducing automata to which, in the abstract, they correspond. This theory was originally developed by Von Neumann (1966), first in terms of a Turing representation and later in terms of tesselation automata. Subsequent relevant work is due to Codd (1968), Burkes (1970) and Vityani (1973) (for tesselation automata) to Ashby (1962), Apter (1966), Baricelli (1962), Von Foerster (1960), Weiner (1962), McCulloch (1966), Fogel et al. (1966) (for general and statistical models) and to Myhill (1970) and Loefgren (1972) for Turing representation. Loefgren's contribution is especially germane to the present discussion.

2. To leave the reader under no illusions, the premises for my theory of conversations are chiefly culled from experience with computer simulated reproduction/evolution in systems having minimal specific structure (Pask 1959, 1960, 1961, 1969, 1974). The authors referenced have similar findings and it is fair to say that a great deal of firm knowledge exists regarding populations of reproductive automata; some of it gleaned from simulation, some by abstract manipulation. For example, reproducibility (in a domain), specificity or compatibility, and evolution are ubiquitous.

The following comments concern the provability or otherwise of certain theorems which guarantee that such events must take place in any appropriate system.

3. With Loefgren, we take the reproduction of a function (or, in general, a relation) π_0 to be

$$\pi_1 (\pi_0) = (\alpha, \beta, \ldots\ldots \pi_0)$$

where π_1 may either be interpreted as an abstract machine or automaton that models π_0 via a series of steps $(\alpha, \beta, \ldots.)$ or as an explanation that proves π_0 starting with the axiom(s) α. In the theory of strict conversations π_0 is identified with a relation in the conversational domain and π_1 with a procedure Proc 1 which is a concept of R_0 if and only if it reproduces R_0; that is Proc 1 (R_0) = $(\alpha, \beta, \ldots. R_0)$. The construction ending in R_0 is called a

modelling operation, or, synonymously, an execution sequence of operations elicited by a command. An L description of this sequence is written D $(\alpha, \beta, \ldots . R_0)$. It constitutes an explanation of R_0 and might have been elicited directly (as an L expression ending in $D(R_0)$) by an explanation demanding question.

4. As a matter of convenience, any reproduction of a reproductive agent is usually distinguished to avoid self reference. That is

$$\pi_2 (\pi_1) = (\gamma , \delta , \ldots . . \pi_1)$$

when π_1 operates on a domain containing π_0 but not π_1 and π_2 operates on a domain containing π_1 but not π_2. So, for example, Proc 2 (Proc 1) = $(\gamma, \delta, \ldots .$ Proc 1) is equivalently cited as the reproduction of a concept Proc 1 or as a memory of R_0.

The requisite constraints are imposed by stratifying the conversational object language, L, into levels of control. This expedient is used extensively in the sequel and it is important to recognise that the strata are quite arbitrary; they do not image some property of real things. For instance, it is possible and permissible to choose rules that admit entities that are symbiotically self-reproducing, like

$$\pi_1 (\pi_2) = (\alpha^*, \beta^*, \ldots . . \pi_2)$$
$$\pi_2 (\pi_1) = (\gamma^*, \delta^*, \ldots . . \pi_1) \qquad \text{Repro 1.}$$

or even (Loefgren's complete self reproducibility)

$$\pi (\pi) = (\mu , \eta , \ldots . . \pi) \qquad \text{Repro 2.}$$

The organisations Repro 1 and Repro 2 are self-reproducing systems that image P Individuals.

The advantages and disadvantages of stratification have been discussed at length by Gorn. Stratified control of one process by another pays dividends insofar as the processes in question can be unambiguously observed and described. However if the stratification is complete, then no self-referential process is allowed. In contrast, unstratified control (Gorn's (1966) name for a linguistic structure that does accommodate Repro 1 and possibly Repro 2) though nearer to characterising natural language, does

not ensure precise and unequivocal observation. The theory of strict conversations is based upon a compromise; to some extent due to a remark of Gorn's (1966) that the 1966 version of the theory was largely couched in terms of stratified control and, because of that, was beset by certain unnecessary limitations.

5. A strict conversation is a P Individual the execution of a symbiotically self-reproducing system (in the sense of Loefgren's development) with respect to a conversational domain R (a surrounding in Loefgren's sense though R, in fact, may include other reproductive systems).

The requisite mathematical development is chiefly contained in one paper (Loefgren 1972).

(a) The existence of productive and self-productive machines in a domain (alias a surrounding) R is shown by Theorem 1.5, p.361.

(b) The existence of a symbiotically self-reproducing system is shown by Theorem 11.5, p.361.

(c) Any such system has its symbionts as factors in R.

(d) The minimal factor is self-productive (but not reproductive in R).

(e) Factoring may be represented, in dynamic models (such as the icons) by reducing each reproductable loop (one, at least, for each level) either into commands/executions or questions/replies.

(f) The existence of reproductive automata that perform other operations, apart from reproduction, is given in Theorem 11.9, p.364.

(g) The existence of indefinitely many automata produced by one automaton in a surrounding (hence, learning or evolution) is shown by Theorem 11.10, p.365.

(h) The concept of P/P compatibility is evident from the preamble to these theorems (for example, change the characterisation of the surrounding). It is more specifically exhibited, in the context of genetic P/P compatibility, by Vityani (1973).

(i) Any self-reproducing system (of this type) contains a self description. Hence, whilst stable, it may be addressed and named as a unit.

6. The Turing representation of reproduction constitutes an abstract or idealised image. It may be generalised and actualised by invoking the ideas of general programming schemata (Burkes, in Burkes 1970 or modular computation (Halland, in Burkes 1970). Perhaps the most fruitful generalisation is obtained by noting that

a class of 'Fuzzy algorithms' can be defined (Zadeh 1968, 1973) and that there also exist Fuzzy Turing Algorithms (Turing Machines) Santos (1970), Zadeh (1968). Using this fact, it is possible to construct Fuzzy reproductive algorithms, also, and some of them will be noted in the next volume. Fuzzy algorithms may be executed in several ways; by selecting one assignment at each computational step or probabilistically (doing, with specified probability, just one of the operations permitted by the Fuzzy specification at each step, or assigning a value according to a probability distribution). Again, the Fuzzy algorithm may be executed in parallel (doing all the operations or assignments simultaneously at each step). Insofar as the parallel activities do not conflict, no qualification is needed; if they do, then the conflict must be resolved by communication (through a supervisor or otherwise) and the resulting process is concurrently executed. It is shown, in the next volume, that it is possible to construct a processor, with a parameter attached to it, which interprets and executes the same Fuzzy algorithm in any of these ways depending upon the parameter value.

The procedures called Concept and Memory are, in fact, Fuzzy reproductive algorithms under concurrent execution.

7. Each of these reproduction paradigms (even the Fuzzy one) entails synchronous execution. That is, the computation steps are ordered and this order is placed in correspondence with the beats of a processor clock. Hence, a Fuzzy instruction like

'If X is small then Y is large else go to P' is acceptable, but

'If X is small then Y is large else go to near P' is not. 'Near P' has no practical meaning under the restriction that execution steps are well ordered.

But, apart from elegance and clear formulation, there seems to be no reason why the execution steps (or the coordinates of the Fuzzy relations defined by conditional imperative statements and brought about when the algorithm is executed) should not be Fuzzified (a legitimate operation within Fuzzy Set theory). In the next volume we show that Fuzzy instructions such as 'If X is small then go to near Y else go to near P' are (under certain conditions) quite acceptable; that their inclusion generally desynchronises the execution and that (supposing the conditions are properly satisfied) the local synchronisation induced by execution acts as the 'supervisor', mooted in (6) above, and permits genuinely concurrent execution, without invoking any special executive

organisation. The 'conditions' noted in the last paragraph have to do with descriptions of the process addresses; roughly that neighbouring addresses in the description are allocated to extensionally equivalent processes (that is, processes that compute the same relation in any manner) and that extensionally distinct processes be further apart (in the description).

Certainly, there are reproductive Fuzzy algorithms that are executed with only local synchronicity (this point is demonstrated in the next volume also) and, in general, the procedures called Concept and Memory are taken to be of this kind.

8. The final requirement, at a theoretical level, concerns interpretation rather than mathematical form. Not only is the notion of memory equated with reconstructability and thus reproduction (a fairly uncontentious idea), but the entire theory is founded upon an explicit identification of the basic system property 'stability' with 'reproducibility', and 'is stable' with 'is being reproduced'. If taken glibly, and at its face value, the proposition is no more than commonsense (psychological commonsense, at any rate). That interpretation is quite acceptable. But the reader may care to ponder upon the possible consequences of a broader interpretation. Stability is the primary feature of a system (alias a relation) by virtue of which the participant is aware of it. If a system is stable, then it may be noticed by a participant and the system (that is, the relation) is the entity noticed. On reversing the argument of course, the system may be constructed or modelled. But stability, in this context, implies only a form of invariance (we have cited temporal invariance and continue to do so; but the invariance need not be temporal). More generally, the system or relation is an invariance attached to a process; it marks the locus of a stability (invariance) where the participant would otherwise sense a void; an amorphous surrounding or a sense of ongoing.

The logic of commands (imperative logic; part of Von Wright's (1963) deontic logic) and the logic of questions (especially those that call for an explanation in reply) are quite distinct from assertoric logic. Rescher's (1967) logic of commands is, so far as I know, the only detailed essay in this field.

One crucial aspect of Rescher's argument is the notion of command termination; a command addressed to some recipient under conditions which render its execution both appropriate and possible either is or is not terminated. It is terminated if the addressee has done whatever has to be done (given the initial conditions and the command). This idea is essential in talking of inference, implication and the like; also in developing the concept of command coverage. We use these ideas ourselves; but since the present approach is operational and psychological the bias is quite different. In particular, though commands are issued and received in the conversational language L (in terms of which they are "obeyed" and lead to "satisfaction" (of relations) and "agreement" (between the commander and the addressee) the constructs "termination" and "coverage" (as well as others, derived from them) have the status of L^* or metalinguistic statements about the conversation; consequently such statements are made by an external observer, only.

A further difference between Rescher's logical theory and the theory of conversations is as follows: Rescher notes that a command either is or engenders a procedure but confines his attention to serial procedures or programs. Further, he need not stress the distinction between "is" and "engenders" because command theory is concerned with logical entities (human beings or machines as the case may be) and it is quite legitimate, for this purpose, to imagine that these entities "stop", either after the execution of a command, or before a command is received. We can afford neither the restriction to seriality nor the idea of stopping; conversation theory is about the psychological mechanisms underlying the construction and execution of procedures which (special cases apart) are definitely not serial programs; moreover, one tenet of the theory is that human beings are made

to and are bound to learn (i.e. to construct procedures), as a result of which stasis is unacceptable. If a command is issued under conditions which constitute a domain where some procedure (alias concept) is applicable, then, if the command is obeyed this procedure is applied. If there is no applicable procedure in the addresses repertoire, then efforts are made to construct and reconstruct a procedure, that can be obeyed.

1. *Simple commands and question forms.* A command is issued, on authority, to an addressee, Z; it is to be executed if and only if certain preconditions are satisfied and it either specifies an action to be performed or a further condition to be brought about (Rescher's "state achievement commands"). We are chiefly concerned with "state achievement commands", and represent them in the form.

$$\underline{\text{Comm}}_Z\, i = \langle Z!\ \text{Bring about } R_i/\text{Precondition} \rangle$$

The precondition is complex and invariably consists in the statement of an occasion n or an instant t_n when the command is to be obeyed, together with one or more other relations that must hold at the occasion or instant concerned. Unless it is essential to the argument, the occasion or instant is suppressed notationally, but it forms part of each precondition. Thus

$$\underline{\text{Comm}}_Z\, i = \langle Z!\ \text{Bring about } R_i/R_j, R_k \text{ Hold} \rangle$$

or, tersely

$$\underline{\text{Comm}}_Z\, i = \langle Z!\ R_i/R_j, R_k \rangle$$

If the precondition is the constraint upon a modelling facility

$$\underline{\text{Comm}}_Z\, i = \langle Z!\ R_i/M \rangle$$

Obedience consists in one or many stages of modelling; hence, the command might be replaced as an injunction to carry out a modelling operation (Exec); either with specific conditions given at the outset.

$$\underline{\text{Comm}}_Z\, i = \langle Z!\ \underline{\text{Exec}}\ i/R_j, R_k \rangle$$

or given only the constraints in the modelling facility

$$\underline{\text{Comm}}_Z\, i = \langle Z!\ \underline{\text{Exec}}\ i/M \rangle$$

Since an explanation (Expl) is interpreted as a description of a model, the "command to explain" is a corresponding "how" or "why" question asked of Z and this interrogative form is simply

$$\underline{\text{EQuest}}_Z\, i = \langle Z!\ \underline{\text{Expl}}\ i/R_j, R_k \rangle = \langle Z?\ R_i/R_j, R_k \rangle$$

(explain how R_i may be modelled, if R_j and R_k hold) or even

$$\underline{\text{EQuest}}_Z\, i = \langle Z!\ \underline{\text{Expl}}\ i/R_i \rangle = \langle Z?\ R_i/R_i \rangle$$

(explain how R_i was modelled, given that R_i holds).

It is possible to regard the addressee, Z, as a processor $(\alpha, \beta, \ldots)$ with undetermined repertoire or as a P Individual $(A, B, \ldots)$ with repertoire π_A, π_B, executed in some (generally undetermined) processor. We opt for the latter possibility (setting Z = A, though similar comments apply if Z = B and B receives commands from A) and interpret obedience (by A) as follows. On receipt of $\underline{\text{Comm}}_A\, i$ (or any derived interrogation), if A obeys the command, then A searches π_A for a $\underline{\text{Proc}}_A\, i$ which may be assigned initial values in the domain specified by the precondition; if such a procedure exists then A applies it and manifests Exec i (or Expl i).

Rescher calls commands of this type (in contrast to chains or sequences of commands) simple. Under the chosen interpretation (Z = A), simple commands are the base commands of conversation theory. A base command $\underline{\text{Comm}}_A\, i$ issued for receipt on occasion n is terminated if and only if the following L* (or metalinguistic) statement is true "$\underline{\text{Comm}}_A\, i$ was issued by B to A at t_{n1} and A brought about R_i at some later instant in the occasion (t_{n2}) given the preconditions stipulated".

2. *Command termination.* The idea of termination has tenure in the external observer's metalanguage, L*, not in L. Thus, A obeys (or disobeys) B, which is an L construct: A and B agree (or disagree) that R_i is satisfied, again an L construct. If an external observer has an L* description of D(R) and if A and B are engaged in a strict conversation, he may be able to equate A, B agreement with the termination of $\underline{\text{Comm}}_A\, i$. Further the following formal comments make sense to this external observer for (base) commands (written $\underline{\text{Comm}}_A\, 1\ \underline{\text{Comm}}_A\, 2$); namely,

(a) $\underline{\text{Comm}}_A\,1$ covers ($\gg$) $\underline{\text{Comm}}_A\,2$ if and only if the termination of $\underline{\text{Comm}}_A\,1$ implies the termination of $\underline{\text{Comm}}_A\,2$.

(b) The base command may be contextually qualified by an L statement (S) attached either to the precondition (thus restricting the domain of application of any $\underline{\text{Proc}}_A\,i$ executed), or to the requirement (thus restricting how R_i may be brought about, say to $\underline{\text{Proc}}_A\,ij$). Contextual qualification of a command is thus expressed

$$\underline{\text{Comm}}_A\,i\,(S) \;=\; <A!\ R_i/\text{Precondition} \wedge S>$$

(c) If it happens that $\underline{\text{Comm}}_A\,1\,(S)$ covers $\underline{\text{Comm}}_A\,2\,(S)$ (or that $\underline{\text{Comm}}_A\,1$ covers $\underline{\text{Comm}}_A\,2$ in the context of S) then this contextual coverage is written

$$\underline{\text{Comm}}_A\,1 > \underline{\text{Comm}}_A\,2 \quad \text{Given S.}$$

3. *Decomposition and synthesis.* The following developments (all culled from Rescher) also have meaning to an external observer (Precon standing for a command precondition).

(1) A (Base) command $\underline{\text{Comm}}_A\,0$ may be decomposed into constituents $\underline{\text{Comm}}_A\,1$, $\underline{\text{Comm}}_A\,2$ where $\underline{\text{Comm}}_A\,1 = (A_1!\ R_i/\underline{\text{Precon}}\ 1)$ and $\underline{\text{Comm}}_A\,2 = (A_2!\ R_2/\underline{\text{Precon}}\ 2)$ and $\underline{\text{Comm}}_A\ 0 = (A!\ R_0/\underline{\text{Precon}}\ 0)$ if either

(a) $A_1 \cup A_2 = A$ and $R_1 = R_2$ and $\underline{\text{Precon}}\ 1 \equiv \underline{\text{Precon}}\ 2 \equiv \underline{\text{Precon}}\ 0$

or

(b) $A_1 = A_2$ and (if both R_1, R_2 hold, then R_0 holds) and $\underline{\text{Precon}}\ 1 \equiv \underline{\text{Precon}}\ 2 \equiv \underline{\text{Precon}}\ 0$;

or

(c) $A_1 = A_2 = A$ and $R_1 = R_2 = R$ and ($\underline{\text{Precon}}\ 1 \cup \underline{\text{Precon}}\ 2$) implies $\underline{\text{Precon}}\ 0$.

A decomposition of $\underline{\text{Comm}}_A\,0$ is a list of commands derived whilst obeying these rules.

The termination of $\underline{\text{Comm}}_A\,0$ implies the termination of all commands in any of its decompositions.

(2) The logical connectives corresponding to "conjunction" and the "exclusive or" of assertoric logi are fusion and disjunction. They are specified for commands issued to the same addressee and (in the case of disjunction) having the same precondition.

(a) $\underline{\text{Comm}}_A\,0$ is the fusion of $\underline{\text{Comm}}_A\,1 = (A!\ R_1/\underline{\text{Precon}}\ 1)$ and of $\text{Comm}_A\ 2 = (A!\ R_2/\text{Precon}\ 2)$ written $\text{Comm}_A\ 1 \asymp \text{Comm}_A\ 2$ or "χ $\underline{\text{Comm}}_A\ 1$, $\underline{\text{Comm}}_A\ 2$) if and only if $\underline{\text{Comm}}_A\ 0 = (A!$ (bring about both R_1 and R_2)/$\underline{\text{Precon}}\ 1 \cup \underline{\text{Precon}}\ 2$).

(b) Two forms of disjunction $\underline{\text{Comm}}_A\,1 \vee \underline{\text{Comm}}_A\,2$ and $\underline{\text{Comm}}_A\,1 \;\bar{\vee}\; \underline{\text{Comm}}_A\,2$ are considered by Rescher; (since he conceives the command, qua program as undergoing serial execution he does not examine others). Of these $\underline{\text{Comm}}_A\,1 \;\dot{\vee}\; \underline{\text{Comm}}_A\,2$ is "either $\underline{\text{Comm}}_A\,1$ or (exclusively) $\underline{\text{Comm}}_A\,2$ whichever you choose".

The other form $\underline{\text{Comm}}_A\,1 \;\bar{\vee}\; \underline{\text{Comm}}_A\,2$ means "either $\underline{\text{Comm}}_A\,1$ or (exclusively) $\underline{\text{Comm}}_A\,2$ depending upon circumstances", which is interpreted as "depending upon a condition holding", or depending upon "a modification or qualification of the precondition".

(3) Due to the connotation of "choice" in conversation theory, these disjunctions may be regarded uniformly and, in any case, either termination of $\underline{\text{Comm}}_A\,1 \;\dot{\vee}\; \underline{\text{Comm}}_A\,2$ or of $\underline{\text{Comm}}_A\,1 \;\bar{\vee}\; \underline{\text{Comm}}_A\,2$ implies the termination of $\underline{\text{Comm}}_A\,1$ or of $\underline{\text{Comm}}_A\,2$ or both.

(4) To derive a further meaning for disjunction (more like inclusive disjunction) we need Rescher's concept of command pre-emption and preclusion (wedded in an appropriate fashion, to our ideas of instant and occasion).

(5) At an instant t_n, in occasion n, $\underline{\text{Comm}}_A\,1$ preempts/precludes $\underline{\text{Comm}}_A\,2$ if, at t_n, termination ($\underline{\text{Comm}}_A\,1$) is inconsistent with termination ($\underline{\text{Comm}}_A\,2$). The preemption/preclusion applies equally well to logical or local considerations; (for example, that +1 and −1 appear simultaneously in one register). The general case is a set of η_0 commands $\underline{\text{Comm}}_A\ \eta$; $\eta = 1. \ldots \eta_0$ (with the same addressee) so coupled, in execution, that any preemption/preclusion is prohibited ("no conflict", a condition that may depend on the processor, as well as the process). Under these circumstances a command to execute all η_0 of $\underline{\text{Comm}}_A\,\eta$ is written $\underline{\text{Comm}}_A\,1 \;\check{\vee}\; \ldots \;\check{\vee}\; \underline{\text{Comm}}_A\,\eta_0$ or $\overset{(\eta)}{\vee}\ \underline{\text{Comm}}_A\,\eta$; an "all but precluded" disjunction.

(6) There is no difficulty in giving a command to a group of addressees provided there is a common precondition and a common requirement. The group may be addressed (Rescher) in a distributive or collective manner. For example, $\text{Comm}_{A \cup B}\,i$ is a

command addressed to A and B as a *group* (namely the A, B conversation) to bring about R_i, given Precon i. It either means that both of A, B should bring about R_i (the collective sense) or one of them should do so (the distributive sense). The joint injunction is not trivial, even though A and B are commanded to do the same thing, insofar as they have distinct $\underline{Proc}^0$s; that is $\underline{Proc}_A$ l,i is not $\underline{Proc}_B$ k,i though both $\underline{Proc}_A$ l,i and $\underline{Proc}_B$ k,i bring about R_i, if Precon is given.

(7) We interpret any such command in its distributive form as an $\dot{\cup}$ or $\dot{\cup}$ disjunct over addressees (with both R_i and Precon fixed).

"Either A (exclusively) bring about R_i, or B (exclusively) bring about R_i" or, in its collective form, as the $\cup$ disjunct.

"Both A and B bring about R_i by your own (compatible) methods".

Thus $\cup$ ($\underline{Comm}_A$ i, $\underline{Comm}_B$ i) = $\underline{Comm}_{A \cup B}$ i calls for the parallel and independent execution of $\underline{Proc}_A$ l,i and $\underline{Proc}_B$ l,i. No issue of compatibility arises, provided A and B act in isolation until the (same) result, R_i, is achieved redundantly.

(8) The case when A and B are executed in the same processor (α, say) poses no fresh problems provided that isolation is maintained or provided that A and B compute in the same way. Otherwise computational conflict is possible and must be resolved by appropriate preclusions and preemptions.

(9) Consider A. As noted repeatedly, a P Individual may have many factors, that are themselves P Individuals, and are thus open to being addressed. Call the factors (if they exist) A_μ. If the A_μ are distinct in the domain of R_i (when the sprout of a strict conversation is at node i on occassion n) then, if they compute R_i at all, they compute it distinctly. Hence, the index μ picks out values of the index η and a distributive command, addressed to all A_η, may be written either.

$$\underline{Comm}_A\, i = \overset{\dot{\cup}}{\eta} < \underline{Comm}_A\, i\eta > \text{ (no conflict between procedures indexed } \eta)$$

or

$$\underline{Comm}_A\, i = \text{All } A_\mu \text{ in A such that } \mu - \eta ; \underline{Comm}_{\underset{\mu}{\cup}\, A_\mu}\, i$$

(10) We take either of these as the usual meaning of a Base Command addressed to A, noting the trick that conflict free execution depends upon the external observers account of preemption. There is also an internal conflict avoidance mechanism built into the structure of any viable system, R_i. McCulloch (1965) calls it "redundancy of potential command" and Kilmer and McCulloch simulated the action of the mammalian reticular core (using a program, SRETIC, 1968) in terms of this mechanism.

(11) Conversely, there is another interpretation of a base command, namely

$$\underline{Comm}_A\, i = \overset{\dot{\cup}}{\eta} < \underline{Comm}_A\, i\eta >$$

or

$$\underline{Comm}_A\, i = \text{Just one (any one) } A_\mu < A_\mu ! \underline{Comm}_A\, i >$$

(12) We call (9) the strongest or $\ominus$ interpretation of $Comm_A$ i and (11) the weakest or $\oplus$ interpretation.

4. *Command stratification.* Since the conversational language L is stratified (L = L^1, L^0) it is possible to issue commands at either level. The discussion up to this point, though applicable to either level of command, has been phrased in terms of the L^0 base commands, $\underline{Comm}^0_A$ i. L^1 base commands differ insofar as they consist in a requirement to construct a $\underline{Proc}^0_A$ i and a precondition, which is the existence, in π^0_A, of certain other procedures ($\underline{Proc}^0_A$ j,i, $\underline{Proc}^0_A$ k,i . . . .). Thus, an L^1 base command, if issued at occasion n, is

$$\underline{Comm}^1_A\, i = < A ! \underline{Proc}^0_A\, i / \underline{Proc}^0_A\, j \text{ in } \pi^0_A(n) \wedge \underline{Proc}^0_A\, k \text{ in } \pi^0_A(n) \ldots\ldots \wedge \underline{Prim}^1 \text{ in } \pi^1_A(n) > .$$

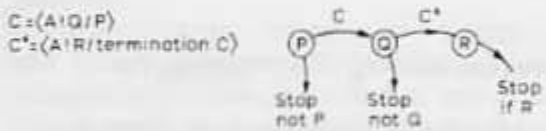
If R_i is in the sprout of a strict conversation, governed by a CET heuristic, this heuristic guarantees, at occasion n, that all clauses in the precondition of $\underline{Comm}^1_A$ i are satisfied.

5. *Standard conditional imperatives.* Commands, as they have been described, act as addressed "If . . . . then" injunctions ("If Precondition . . . . then obtain requirement"). In fact, there is a

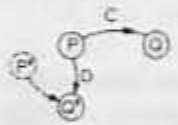
latent "else" clause, applicable if the precondition does not hold. Since command logic tolerates a stop condition, this clause need not be made explicit; but it may be. If it is, $\underline{\text{Comm}}_A$ i becomes an addressed, but otherwise standard, conditional imperative with undetermined "else"; thus

$$\underline{\text{Comm}}_A\,\text{i} = \langle \text{A! } R_i/\text{Precondition} \rangle$$
$$= \text{A! [If precondition then } R_i \text{ else?] .... Eq (1)}$$
$$= \text{A! [If X then Y else U] ..... Eq (2)}$$

Simple command (unconditional)



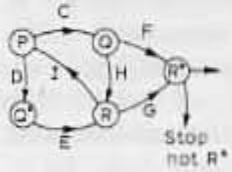
Command sequence



Simple command with explicit conditional

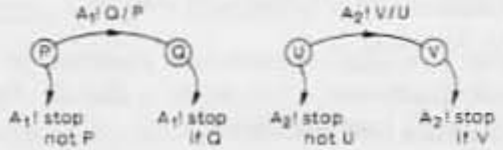
⟨A!Q/P else D⟩ = V(⟨A!Q/P⟩⟨A!Q*/not P⟩) or (alternatively) P* in place of not P

D = ⟨A!Q*/not P⟩ or (alternatively) ⟨A!Q*/P*⟩ (notice P* is not necessarily the compliment of P)



A command program with conditional executions shown as D,H,I and forward sequences C,E,F,G

T = ⟨⟨A_1!Q/P⟩,⟨A_2!V/U⟩⟩ Command chain (T) formed by odering and chain (T*) formed if U,Q by implicative coupling of relations



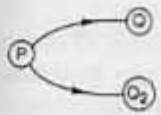
T* = ⟨⟨A_2!V/U⊂Q⟩,⟨A_1!Q/P⟩⟩

Diagram G.1. (Continued on p. 477).

⟨A_1!Q/P⟩ = E_1

(A_2!Q/not P) or (alternatively) ⟨A_2!Q/P*⟩ = E_2

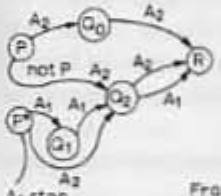
The construction may imply concurrent operation when it tallies with A_1 A_2!Q/P. In our scheme A_1 and A_2 would compute Q differently



Either ∪(⟨A!Q_1/P⟩⟨A!Q_2/P⟩)

or ∩ (⟨A!Q_1/P⟩⟨A!Q_2/P⟩)

or any A_η⟨A_η!Q = (Q_1 ∪ Q_2)/P⟩ (A_1 does Q_1 and A_2 does Q_2)



Fragment of a network (arcs labelled by participants A_η only) open to concurrent ' and * interpretations

Diagram G.1. (continued).

The bracketted part of Eq (2) is written, in general

[If X_1 then Y_1 else if .... X_m then Y_m] ..... Eq (3)

or, by successive application of the conditional and rearrangement

[If X then (If Y then V else U_1) else U_2] ..... Eq (4)

6. *Substitution.* The terms Y, U, V, in these constructions, may be replaced by assignments, command requirements, by an instruction to stop computation (disallowed in conversation theory) or by addresses. In machine idiom, these addresses would locate further instructions (for example "go to index 18" or "do routine 8 returning control after routine 8 is done"). Similarly, the "??" in Eq. (1) is replaceable by "stop" (here inadmissable); by a further command to the same addressee (one form of command sequence, the other being a command issued on termination of the original command); by a command to a different addressee (one form of a command chain, the other form being a command to a different addressee, issued on termination of the original command). Using a pictorial convention which is intelligible in machine idiom,

where conditions are represented as nodes in a graph and operations as labels on arcs, we show these essentially serial cases, in Diagram G (I).

7. *Concurrent command execution.* The general conditional imperative (Eq (3)) represents a function; relating in the simplest case, sets X and Y. A procedure is an expression of this type in which any or all of the terms may be expanded as in Eq (4) and in which assignments necessary for execution have been made.

If X and Y are fuzzy sets (Zadeh 1968) the resulting (Fuzzy) conditional imperative may be interpreted as a Fuzzy Relation (Zadeh 1973). In computational terms, (Appendix F, Sections 6 and 7) this circumstance, which is ubiquitous, gives rise to concurrent computation. The execution of the procedure is itself a graph (call it a command graph, rather than a command sequence) and fragments of such structures are shown in Diagram G1 (II).

As noted in Appendix F, sections 6 and 7, a Fuzzy Algorithm (or a non-deterministic program as a special case) is open to several possible interpretations in a sufficiently capable processor. In particular:

(a) The weakest or $\ominus$ interpretation of a base command gives rise to a serial execution (if one exists).

(b) The strongest or $\oplus$ interpretation of a base command gives rise to a concurrent execution.

(c) The L^0 description of what may be done, $D^0(R_i)$ is a command graph with preclusions/preemptions to avoid conflict inserted by an external observer or constructed by a subject matter source.

(d) If the modelling facility is one clocked, then models are serial representations, corresponding to any one command sequence in a command graph.

8. *The participant as a system that learns.* One basic tenet of conversation theory is that any participant is made, or organised, to learn. This point is expressed in command nomenclature, as follows.

$$\underline{\text{Comm}}^0_A\, i = < A!\ R_i/X \text{ else? } >$$

The "?" must be substituted by something other than "stop". It may be substituted as proposed in the last sections (for

example, by a further L^0 command) but, if not (and ultimately always) by $\underline{\text{Comm}}^1_A\, i$. That is

$$\underline{\text{Comm}}^0_A\, i = < A!\ R_i/X \text{ else } \underline{\text{Comm}}^1_A\, i >$$

similarly (omitting occasion designators, but assuming a particular occasion),

$$\underline{\text{Comm}}^1_A\, i = < A!\ \underline{\text{Proc}}^0_A\, i\ /\ \{\ \underline{\text{Proc}}^0_A\, ij\ \}, \underline{\text{Prim}}^1_A > \text{else } \underline{\text{Comm}}^0_B\, i$$

If $\underline{\text{Comm}}^1_A\, i$ is terminated, then $\underline{\text{Comm}}^0_A\, i$ can be obeyed (and $\underline{\text{Comm}}^1_A\, i$ would not have been issued unless, previously, R_i could not be brought about due to the lack of a $\underline{\text{Proc}}^0_A\, i$). If $\underline{\text{Comm}}^1_A\, i$ is not terminated then a command is issued to B.

9. *Coverage relations between commands at level L^0.* A command sequence that achieves the same result (and has the same precondition) as a base command is more complex and refined than the base command itself: hence, it covers it. So, in general, does a command graph, permitting concurrent but also detailed computation. Hence, even if $\underline{\text{Comm}}^0_A\, 1 \gg \underline{\text{Comm}}^0_A\, 2$ (as they may, under circumstances to be specified in the next section) it does not follow that the command graphs corresponding to the base commands (and described by $D^0(R_1)$ and $D^0(R_2)$) also satisfy the coverage relation. In fact, it is possible (as well as usual) for subgraphs of $D^0(R_1)$ and $D^0(R_2)$ to exhibit the reverse coverage relation, and thus to contravene the base ordering. For example, if x is a command graph (or command sequence) in $D^0(R)$ and y in $D^0(R_1)$, it is possible that $y \gg x$ even though R_2 is superordinate to R_1.

10. *Coverage relations between commands at level L^1.* Recall that $D^1(R)$, is an L description of what may be known (in contrast to $D^0(R_i)$, a description of what may be done). $D^1(R)$ may incorporate conjunctive or disjunctive substructures (replicated in Diagram G2) of which the latter is an essential constituent of any analogy relation.

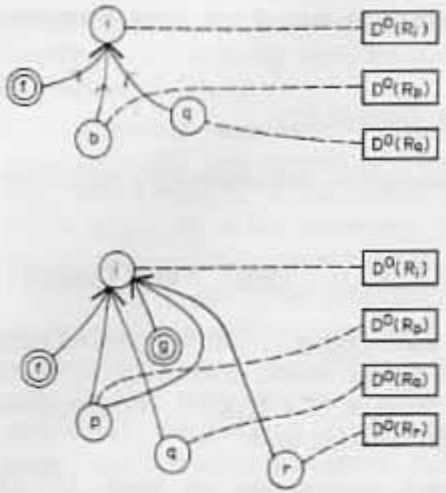


Diagram G.2. A fully conjunctive substructure in $D^1(R)$ and a disjunctive substructure in $D^1(R)$ together with the $D^0(R_i)$ that are attached to them by data links (shown as dotted lines).

(1) For the conjunctive substructure in Diagram G2, and 1 an index over distinct procedures, in π^0_A (n), at occasion n.

$\underline{\text{Comm}}^1_A\,i$ = (A! $\underline{\text{Proc}}^0_A\,1,i$ / $\underline{\text{Proc}}^0_A\,1,p$ in $\pi^0_A(n)$, $\wedge$ $\underline{\text{Proc}}^0_A\,1,q$ in $\pi^0_A(n)$ $\wedge$ ... $\underline{\text{Prim}}^0_A\,F$ in $\pi^0_A(n)$)

The precondition

$\underline{\text{Proc}}^0_A\,1,p$ in $\pi^0_A(n)$ $\wedge$ $\underline{\text{Proc}}^0_A\,1,q$ in $\pi^0_A(n)$ $\wedge$ $\underline{\text{Prim}}^0_A$ F in $\pi^0_A(n)$

is unambiguous and unique.

(2) For the disjunctive substructure in Diagram G2, the L^1 base command has two preconditions, namely

$\underline{\text{Precon}}\,1,i \equiv \underline{\text{Proc}}^0_A\,1,p$ in $\pi^0_A(n)$ and $\underline{\text{Proc}}^0_A\,1,q$ in $\pi^0_A(n)$ and $\underline{\text{Prim}}^0_F$ in $\pi^0_A(n)$.

$\underline{\text{Precon}}\,2,i \equiv \underline{\text{Proc}}^0_A\,1,p$ in $\pi^0_A(n)$ and $\underline{\text{Proc}}^0_A\,1,r$ in $\pi^0_A(n)$ and $\underline{\text{Prim}}^0_A\,G$ in $\pi^0_A(n)$.

Hence, there are really two L^1 base commands, namely

$\underline{\text{Comm}}^1_A\,1,i$ = $<$A! $\underline{\text{Proc}}^0_A$,x,i / $\underline{\text{Precon}}$ 1,i$>$; x = {1}.
$\underline{\text{Comm}}^1_A\,2,i$ = $<$A! $\underline{\text{Proc}}^0_A$,y,i / $\underline{\text{Precon}}$ 2,i$>$; y = {1},
where $x \neq y$

(3) Suppose that both preconditions are satisfied, at an occasion n, in a strict conversation, then $\underline{\text{Comm}}^0_A\,i$ (actually having the two conditional forms just exhibited), is necessarily issued also during this occasion and it may be given two distinct interpretations; a weakest, or $\ominus$, interpretation and a strongest, or $\oplus$, interpretation.

(4) The construction of $D^1(R)$ guarantees that if A receives and obeys $\underline{\text{Comm}}^1_A$; at occasion n when at least one of the preconditions is satisfied (perhaps both) then even if he gives the $\ominus$ interpretation to this L^1 command there is no preclusion which prohibits giving the $\oplus$ interpretation to $\underline{\text{Comm}}^0_A\,i$. Of course, the $\underline{\text{Proc}}^0_A\,1,i$ built by A are almost certain to differ according to which interpretation A does give to $\underline{\text{Comm}}^1_A\,i$. But the relations brought about by any $\underline{\text{Proc}}^0_A\,1,i$ are extensionally equivalent to those brought about by any $\underline{\text{Proc}}^0_A\,k,i$ (the isomorphism asserted by $D^1(R)$; namely $R_i \Leftrightarrow (F\,(R_p, R_q) \Leftrightarrow (G\,(R_s, R_r))$.

(5) In other words R_i is functionally the 'same' relation in the context of $D^1(R)$ however it is learned and subsequently satisfied.

For example, suppose A is learning shopkeeper's accountancy (a relation R_{head} in $D^1(R)$) and is required to perform the usual arithmetic operations in order to exercise that cluster of skills, so, R_i is, for example, division of real numbers.

A might have learned division in many different ways (supposing node i surmounts a disjunctive substructure in $D^1(R)$). A may, as a result, be able to divide by one or many different methods (for example, by long division, by abacus methods, and so on). But, so far as the condition of cyclicity and subsequent reproducibility is concerned, it does not matter in what way he learned division, and further, all methods of division are said to be equivalent, in the context of accountancy skills as their knowable content is represented in $D^1(R)$.

It should be emphasised that these statements are very specific. It is neither asserted nor believed that A's interpretation of $\underline{\text{Comm}}^1_A\,i$ (as $\ominus$ or $\oplus$) is immaterial in general, or that it has no

influence on A's psyche, or learning rate, or his immunity, in contexts other than $D^1(R)$, to operations that interfere with division.

On the contrary, all of these general properties are influenced if A is required to interpret L^1 commands in a particular manner. But the word 'influence' is used guardedly; for, although it is possible to insist upon a $\oplus$ interpretation if otherwise a $\ominus$ interpretation might have been given, there is no obvious way of insisting that A chooses one learning method ($\ominus$) if A opts for $\oplus$ and $D^1(R)$ is disjunctive.

(6) For the conjunctive substructure under node i, L^0 base commands to bring about relations tagged by nodes subordinate to R_i belong to a decomposition of $\underline{\text{Comm}}^0_A$ i. Thus, they are strictly covered by $\underline{\text{Comm}}^0_A$ i. So in Diagram G2, the following metalinguistic (L^*) statements are true.

$$\underline{\text{Comm}}^0_A \text{ i} \gg \underline{\text{Comm}}^0_A \text{ p}$$
$$\underline{\text{Comm}}^0_A \text{ i} \gg \underline{\text{Comm}}^0_A \text{ q}$$

In particular if these commands refer to the same modelling facility, and the preconditions of the same universe,

$$\underline{\text{Precon}} \text{ i} \equiv (\underline{\text{Precon}} \text{ p} \wedge \underline{\text{Precon}} \text{ q})$$

(7) The same comments apply to any one kernel of a disjunctive node but not, as a rule, to all kernels. So, in the disjunctive substructure of Diagram G1,

$$\underline{\text{Comm}}^0_A \text{ i} \gg \underline{\text{Comm}}^0_A \text{ p}$$

and either

$$\underline{\text{Comm}}^0_A \text{ i} \gg \underline{\text{Comm}}^0_A \text{ q} \text{ or } \underline{\text{Comm}}^0_A \text{ i} \gg \underline{\text{Comm}}^0_A \text{ r} \text{ or both}$$

$$\text{Further } \underline{\text{Precon}} \text{ i} \equiv (\underline{\text{Precon}} \text{ p}) \wedge (\underline{\text{Precon}} \text{ q} \vee \underline{\text{Precon}} \text{ r})$$

(8) However, if A engages in a strict conversation regulated by a CET heuristic, there is, in addition, a well-defined contextual coverage for each occasion n. The base command for the head node contextually covers all others; any superordinate contextually covers all of its subordinates; and the contextual L statement (S) is a description of A's learning strategy up to the nth occasion.

11. *Intensionality.* It should be stressed (as suggested by Diagram G2) that the coverage relations of base commands $\underline{\text{Comm}}^0_A$ i, $\underline{\text{Comm}}^0_A$ j, neither necessarily nor usually extend to the command sequences or the command graphs, $D^0(R_i)$, $D^0(R_j)$; since these sequences and graphs cover the base commands. In order to establish relations between commands (or the procedures they invoke) at this level, we must pass from criteria of extensional equivalence (two procedures, on execution, bring about the same relation) to criteria of intensional or procedural equivalence (the procedures have the same, or equivalent, forms).

Several quite different approaches are available. The following possibilities are not exhaustive.

(a) Represent the command graphs, in a canonical form, as Petri Nets (Petri 1965, Holt 1968, Dienes 1972) and compare isomorphic subnets.

The mathematical theory of Petri Nets is still incomplete. Since his paper, interpreting Petri Nets as computing systems, Holt in concert with Commoner (1972), has refined two cases of primary concern to the purist; namely, "occurrence systems" (which admit parallel computation, but no interesting interaction) and the converse, non-parallel case, which is an abstracted finite automaton. Though both cases are fully explored, neither is sufficient for the present purpose.

(b) If the relations expressed in $D^0(R_i)$ are all functions then a combinantonic network (Barralt-Torrijos and Chiaraviglio, 1971, Chiaraviglio, Poore and Barralt-Torrijos 1972, and Barralt-Torrijos 1973) is a general representation.

(1) Consider the least specific meaning an observer might attach to an M Individual i.e. the minimal M Individual. It is a spatially localised physical or biological apparatus, a brain or a computing machine of any kind whatsoever, that does one or more (unspecified) operation(s) when one or more (unspecified) condition(s) are satisfied, together with the following requirement: the apparatus does (unconditionally) the operations needed to make it persist (those operations needed to reproduce it; to render it stable). Thus the minimal M Individual is a physically embodied but non-specific conditional imperative. To give it a less high sounding, but more illuminating, title; it is a processor, specified without reference to the indefinitely large number of procedures it might execute. If presented with a procedure that nominates the missing operation(s) and condition(s) the processor "gives an imperative interpretation to" i.e. it "executes" the procedure.

In addition, the M Individual predicates (and makes assignments of values. For example, the mapping between a semantic domain or universe and the initial values of variables.) This function is relegated, in computer design, to a human being who uses the machine; hence, some of the processor examples furnished later must be qualified by noting that a general purpose computing machine (which is designed not to predicate, since assignments are directly or indirectly determined by the user) becomes an M Individual only when augmented by a human being. The really crucial point is as follows.

It is stressed, in several places, that a procedure does not, of itself, predicate (for example, it is emphasised that the predicative operation of "field iterate", employed in deriving the entailment mesh for a subject matter, only "simulates" the act of predication). A procedure may (and as a rule does) give permission for predication and it does, very often, qualify or restrict the admissible forms of predication (even pointing out one of them only, as legitimate). But it does not do the predication. In contrast, an M Individual does; but unless it is constrained in some manner (for example, by a qualification or restriction contained in a process under execution) it predicates unspecifically. That is, if a

procedure is executed in this processor it is interpreted with respect of any or all possible domains to which it is applicable. This statement is almost trite if we have chunks of nervous system or brains in mind; it is thoroughly irrelevant to general purpose computers designed to work otherwise. A useful bridge is established between these extremities by a philosophical consideration of chemical computing systems, where predication is obviously a propensity of organised fabric; in this respect, the reader may find some early papers (Pask 1959, 1961) of interest.

The essential constraints upon the operation of an M Individual as a processor are to do with order and storage (the latter consists in a specially restricted case of predication). In essence, the processor orders an injection of the negentropy (Brillouin 1965), needed to convert the pattern (latent in a procedure, for example) into potential information; that is, to give instructions an imperative (Do them) interpretation and permissions a subjunctive interpretation (compute a relation).

It is worth considering the generality of this idea, if only because the instances we cite are chosen, for clarity's sake, as pedestrian and specialised.

The most familiar example of negentropy injection, used by Brillouin, is reading a paper tape (a pattern) by means of the negentropy furnished through the energy in a light beam. But energy supply, on its own, is insufficient. The energy inflow (and thus the injection of negentropy) is ordered, either by moving the tape, or by moving the light beam; further, these motions must correspond to the inscription order on the tape. If the tape code positions are identified with the ordering of instructions in a serial program inscribed on a storage medium (their numbering, modified by 'go to' and 'Do' statements) then the light beam movement corresponds to a shift register driven by a clock. On the whole, we use the notion of 'clock' in this sense (as a device for injecting negentropy) and a Shift register (as its cumulative beat counter) but point out that other kinds of scan are possible and commonplace. It is even possible to build systems in which negentropy is injected where and when it is required (the chemical computers, for example); not only the many clocked systems cited below, but processors having as many clocks as necessary.

In the sequel it will be supposed that a processor consists in an assembly of clocks, of shift registers and of stores.

(2) All minimal specifications are parodies of reality and, in

common with many others, this particular specification is a little bizarre. Notably, most of the M Individuals we choose to distinguish have specific and inbuilt operations, over and above those required to maintain the processor's integrity. For example, human and animal brains are bound by innate constraints, to operations that appear (on description) as characteristic behaviours. Only some of the brain (most of the human brain, very little of the amphibian brain) is free to act as a processor in the sense of a program space to be inhabited by procedures[27].

(3) The constraints are more easily exemplified in terms of electronic mechanism rather than biology. In these terms, a processor comprises the assembly of clocks, of shift registers and of stores mooted in (1) above.

From a traditional stance, where integer indexing is taken for granted (and, with it, the ideas of one locus control, of one state at once and one clock of) the simplest paradigm is probably a material embodiment (in real fabric) of a universal Turing Machine. The material embodiment of an abstract digital computer, a "programmed machine" (Minsky 1967) or a "synchronous, centrally regulated, programmable, iterative equipment with a Boolean Algebra over its states set" (Chiaraviglio et al. 1972) is more convenient and realistic since in the latter case, several clocks may operate if they are strictly synchronised.

(4) For the technical reasons, noted in Appendix F and G, we cannot restrict our attention to this "simple" class of processor though it is usefully cited as a category for which a great deal of background knowledge is available regarding what happens if an initial state (conditions and operations for the conditional imperative) is specified by a program and if the system is left to run. In addition, it is at least necessary to countenance processors that are able to execute procedures permitting parallel and concurrent modes of computation. For example, combinatonic networks (Chiaraviglio et al. 1971, Barralt Torrijos and Chiaraviglio 1971) combinatory tesselation automata (Roehkasse

[27] "Most" and "little" are used properly, in this context, as demarcating relative sizes, numerosities of neurones, connecting fibres and so on. Hence, these words carry no immediate commitment regarding relative information capacity. If the processor class is restricted, they may do. For example, if the processor is of the type reserved for modelling facilities, then capacity bounds can be established through Bremmerman's limit (Bremmerman 1962, Ashby 1968).

and Chiaraviglio, 1972) some abstract graph automata (Rosensteihal et al. 1972 and Petri Nets, Holt 1968, Petri 1965) are procedures of this type which call for less constrained processors.

(5) Technical considerations apart, it is desirable to make the class of processor as general as possible. The word "simple" in the last paragraph, was intentionally placed between inverted commas. The processor class under discussion at that point is deemed "simple" for chiefly historical reasons that stem from two developments. It happens that computer design (of physical objects, M Individuals, like "abstract digital computers") was initiated by the mathematical discoveries of Markoff, Church, Post and others; in turn, the success of these devices encouraged mathematicians to work on similar lines. The interaction was especially obtrusive inside the embryonic computer industry of the early 1950s. Because these ways of describing and designing machines as string processors became familiar, they are deemed "simple", as, from that point of view, they should be. But there is nothing in electronics to make them so and if hybrid computer designs had won out at that date, as they are doing so today, then the notion of simplicity would be very different. By the same token, but with much greater cogency, it would be unwise to dwell on apparently "simple" processors if the brain is taken as an M Individual (as it mostly is, in the sequel), or, for that matter, any other large physical system such as a polymerising aggregate, a star, or a cyclic biochemical reaction.

(6) The type of M Individual constraint depends upon the specificity with which the processor, common to all such entities, is isolated by an observer. He may, for example, count organisations like compilers and overall executive routines as processor constraints if these normally programmatic objects are immutably bound to the M Individual. Generally, we mean, by M Individuals, whatever constraints the computer scientist refers to as hardware or storage organisation.

(7) With psychological processes in mind, an external observer is likely to M Individuate entities like human brains or, in view of recent work on other organisms, some animal brains as well; for example, Dolphin brains (Lilly 1961, 1965) or Chimpanzee brains, if they are augmented by the apparatus needed to establish meaningful communication (Premack and Premack 1972). But a modicum of caution should be exercised, for reasons of the following kind.

Any functional human brain reacts; for instance, any effective stimulus gives rise to an electrical response in the frontal cortex that images a usually larger response in the appropriate sensory region. But, under these circumstances, there is no evidence that the brain is acting as an L Processor and there is ample circumstantial evidence that it is not doing so. Apparently, the brain acts as an L Processor if and only if it is organised by an orienting process (involving the cerebral cortex, the reticular formation, and the limbic system) into a special state, manifest, in average EEG records, as the "Contingent negative Variation" (Gray Walter 1968). Given this organisation, the brain, qua M Individual, is an L processor. It may execute any P Individual which momentarily, at least, inhabits the brain in question. If executed, this P Individual entertains an L hypothesis and anticipates an outcome; loosely speaking "the brain does so". Under these conditions, as Gray Walter points out, an external observer can legitimately speak of expectation, information, redundancy, equivocation, relevance, and so on with respect to the hypothesis in hand; otherwise, these terms are rightfully used in respect of the observer's constructs only (just as an observer might speak of "information transfer", computed from the entries in a contingency table representing arbitary experimental data).

(8) It would be out of place to pursue the assumptions underlying the commonly taken-for-granted continuity of habitation by a P Individual, that rightly or wrongly, leads to the notion that "Joe" may be placed in one to one correspondence with "Joe's brain". The example is used to teach one simple lesson; namely that M Individuals like brains, which are "obviously" candidate L Processors, do not invariably act in this capacity. On the other side of the coin, M Individuals that are often neglected because they are inanimate (the computing machines and polymerising aggregates mentioned earlier) may often serve as perfectly good L Processors.

(1) L^0 *Questions of type* $\underline{PQuest}^0_A\, i$

Consider the possible or permissible Im Ent Set k, i, (for all values of k). Their union is Im Ent Set i.

Since subgoal = i all nodes j_{rk} in at least one Im Ent Set k,i, are marked understood and for each of these there is a $\underline{Proc}^0_A\, j_{rk}$ in π^0_A (the CET heuristic guarantees that this is so).

If so, A may regard the relations Rj_{rk} as properties, the values of which he can compute (they are the properties of which each $\underline{Proc}^0_A$ jrk may be a property-value test). Thus A may regard R_i as holding between these properties which, at occasion n (when subgoal = i) are designated $\underline{Prop}_A\, i_n$, one to each of the $\underline{Proc}^0_A$ jrk. Using a similar argument with respect to all values of k, construct a list of pseudo properties PProp i, one for each relation named by a node in Ent Set i. Since some of these may not correspond to nodes marked as being understood, on occasion n, A may not have $\underline{Proc}^0_A$ to compute their values. Thus, either

$$\text{Set of } (\underline{Prop}_A\, i_n) = \text{Set of } (\underline{PProp}^0\, i)$$

or

$$\text{Set of } (\underline{Prop}_A\, i_n) \subset \text{Set of } (\underline{PProp}^0\, i).$$

Each $\underline{PProp}^0$ i and each $\underline{Prim}^0$ corresponds to a unary but many valued predicate $\underline{Pred}^0$ i in L^0. Suppose there are z of them. Any set of exclusively disjunctive L^0 expressions in the z distinct $\underline{Pred}^0$ i is a set of alternatives. For example, if $\underline{Min\ Sent}^0$ is the strongest (conjunctive) form in L^0; if s is an object variable in L^0, and if Val is the value of $\underline{Pred}^0$ [28]; then the qth $\underline{Min\ Sent}^0$ is

$$\underline{Min\ Sent}^0\, i,q = (\underline{Pred}^0\, 1,i(s) = \underline{Val},\, 1,i) \wedge \ldots \wedge (\underline{Pred}^0\, z,i(s) = \underline{Val}\ z,i).$$

[28] $\underline{Pred}^0\ \xi,i$ designates a set of elements $s \in S_{\xi i} = \underline{PProp}^0\ \xi,i$; Val, ξ,i designates a disjoint subset $S_{\xi ir}$ of $S_{\xi i}$ where $\cup\, S_{\xi ir} = S_{\xi i}$.

In particular, if some Min Sent^0 iq denotes each x ϵ X and each y ϵ Y (the states of the modelling facility and the operators) then the Pred^0 are, in aggregate, a fine structure family over a universe of discourse. The 'objects' over which s ranges are "states of the modelling facility" or operators. Conversely "states of the modelling facility" are specified in terms of "state variables" that are PProp^0 or Prim^0.

The Min Sent^0 i,q are alternatives and more complex alternatives (or for that matter any other expressions in the Pred^0 i of L^0) can be synthesised as a union of the Min Sent^0 i, q.

Taken in extension R_i is a subset of the product of the PProps; that is

$$R_i \subset \text{PProp}^0\ 1, i \times \ldots\ldots \times \text{PProp}^0\ Z, i.$$

Moreover, from the construction of $D^0(R)$ this is true of the m_k properties of R_i at occasion n (a consequence of the fact that R_i may be derived by any or all of the prescribed methods and that at least one method was used by A if all the nodes in some Im Ent Set k,i, are marked as understood on the nth occasion). Thus, if R_i is again taken in extension, it is also true that

$$R_i \subset \text{Prop}^0\ 1, i_n \times \ldots\ldots \times \text{Prop}^0\ Z, i_n$$

A valid set of alternatives, Alt Set i, is any collection of exclusively disjoined L^0 expressions in the Pred^0 i such that one and only one L^0 expression describes a set in R_i.

For example, considering MinSent^0 i,q (which designate unit sets in PProp^0 1,i $\times$ . . . . $\times$ PProc^0 z,i) a particular collection of alternatives ($\bar{\vee}$ standing for exclusive disjunction) is

$$\text{Alt Set}\ 1, i = \text{MinSent}^0\ i,1\ \bar{\vee}\ \text{MinSent}^0\ i,2\ \bar{\vee}\ \text{MinSent}^0\ i,3\ \bar{\vee}\ \text{MinSent}^0\ i,4\ \bar{\vee}\ \text{MinSent}^0\ i,5$$

provided that one and only one term satisfies R_i or designates a member of the extension of R_i. Call this the correct alternative (for instance, in this case, MinSent^0 i,1). Call any of the alternatives Alter i and the correct one Alter$^+$ i. Since many Alt Set i with the characteristic that there is one and only one Alter$^+$ i might be constructed (either from MinSent^0 i or more complex forms) the lth valid alternative set is denoted Alt Set l,i.

To exemplify these notions imagine (unrealistically) that the Pred^0 i are binary valued; for notational convenience call them a, b, c, d, and suppose them to have Val a = { 1,0 } Val b = { 1,0 }, Val c = { 1,0 }, Val d = { 1,0 }, commenting that, in a general case, the *values* need not be comparable except by auxiliary indexing.

In this miniature system, there are 16 MinSent^0 i for a 4 term relation. Because the values are comparable and binary a relation can be written in a form such as $R_i = ((a \equiv b \equiv d) \vee (a \equiv c \equiv d))$, and, if that is the relation under scrutiny, then it is satisfied by 6 of the 16 MinSent^0 i namely:

	a	b	c	d
1	0	0	1	0
2	0	1	0	0
3	1	0	1	1
4	1	1	0	1
5	0	0	0	0
6	1	1	1	1

So, for example, there is one two membered Alt Set^0 i consisting in the union of all these 6 correct 4 tuples (Alter 1) and of the union of all of the 10 other MinSent^0 i (Alter 2). Another two membered Alt Set^0 i is any (specific) union of the correct 4 tuples (alter 1) and any (specific) union of the others (alter 2). There are, of course, many membered Alt Set^0 i; for example, it is easy to construct one 11 membered Alt Set^0 i, by taking one correct 4 tuple as the Alter$^+$ i and all of the 10 outside the related subset as the other 10 Alter i. BOSS, in fact, is limited to accommodate no more than 8 alternatives as a practical expedient.

Some care is needed when there is more than one Ent Set k,i. Hence, R_i has been chosen to exhibit this possibility. As R_i is specified two Ent Set k,i are possible, namely the nodes of a,b,d and the nodes of a,c,d; for R_i might be known either by $a \equiv b \equiv d$, or by $a \equiv c \equiv d$, or both. The possible correct alternatives are unchanged. Two different categories of mistaken alternatives become evident; namely those that falsify $a \equiv b \equiv d$ and those that falsify $a \equiv c \equiv d$ and those that falsify both.

For example, of the Alt Sets shown below the distinction between correct and other than correct 4 tuples depends upon the

value of one or both or neither of the variables b and c. For Alt Set 1,i; b is irrelevant to this discrimination; For Alt Set 2,i; c is irrelevant. Thus any participant could discriminate the Alter$^+$ of Alt Set 1,i regardless of his possession of a concept for computing the values of b; likewise the discrimination of Alter$^+$ in Alt Set 2,i does not depend upon a concept for computing the values of c.

	Alter$^+$	Alter$^-$
Alt Set 1,i	{ 1 0 1 1 ; 1 1 1 1 }	{ 0 0 1 1 ; 0 1 1 1 }
Alt Set 2,i	{ 1 1 0 1 ; 1 1 1 1 }	{ 0 1 0 1 ; 0 1 1 1 }

Although many points might be made with the help of such an example (about the distance of an Alter i in an Alt Set; in this case a Hamming distance, usually a generalisation of that measure) the crucial issue is that because of the redundancy the $\underline{\text{Prop}}^0$ named by the $\underline{\text{Pred}}^0$ called b and $\underline{\text{Prop}}^0$ named by the $\underline{\text{Pred}}^0$ called c may either be a $\underline{\text{Prop}}^0_A\, i_n$ or a $\underline{\text{PProp}}^0 i$ provided only that one of them is a $\underline{\text{Prop}}^0_A\, i_n$. Moreover, a $\underline{\text{PQuest}}^0_A\, i$ will not, on this account, receive a different Alter$^+$i selection. But A's reasons for selecting it and discriminating between rejected alternatives will differ greatly according to his concepts and are open to investigation over an Alt Set i such as the one shown below.

Alter$^+$	Alter 1	Alter 2
{ 1 0 1 1 ; 1 1 0 1 ; 1 1 1 1 ; 0 1 0 0 ; 0 0 1 0 ; 0 0 0 0 }	{ 0 0 1 1 ; 0 1 1 1 }	{ 1 1 0 0 ; 1 1 1 0 }

Alter 3	Alter 4	Alter 5
{ 0 1 1 0 ; 1 0 0 1 }	{ 1 1 0 0 ; 0 1 0 1 }	{ 1 0 1 0 ; 0 0 1 1 }

(2) *L^1 Questions of type $\underline{\text{PQuest}}^1 i$*

It is only necessary to repeat the construction already furnished for the $\underline{\text{PQuest}}^0_A i$ with the following variations on a common theme.

(a) The universe of interpretation is no longer the modelling facility alone but the set of nodes or place holders for relations R_i. The index, i, is an L^1 object variable that names node i and consequently R_i.

(b) Unary and many varied L^1 predicates or *descriptors* (symbolised as $\underline{\text{Pred}}^1 i$) replace the $\underline{\text{Pred}}^0 i$ in all of the preceding expressions. By edict, the descriptors of $D^1(R)$ form a fine structure family; but this fine structure family is not full; (the last condition ensures the possibility of pointing to a unit class in the L^1 descriptor that currently contains no node, and thus of extending R in the evolutionary mode when R = R(n)).

1. *Relational operators*

The requisite operations to realize Field (R) are the following

Successor (i) = i+1 , with base 1, $i \in I$

Coordinate index name = $n \in I$

Specification of Set = $S_1 = \{S_i\}$

Value of coordinate (i) = $S_{ij} = \{S_{ijk}\}$ and such that:

$S_{ij} \subset S_i$, $\cup_{j=1}^{j=m} S_{ij} = S_i$, $\cap_{j=1}^{j=m} = \phi$ (the empty set)

Cartesian product $(R_1 \times R_2 \times \ldots \times R_n) = C = \{<r_1, r_2, \ldots, r_n>:$
$(r_1 \in R_1) \wedge (r_2 \in R_2) \wedge \ldots \wedge (r_n \in R_n)\}$

Index name for n-tuple $j \in J$ in Cartesian Product S or (later) in relation R in S, defined over values of one coordinate (in Codd's nomenclature, the primary key).

Indices of Field (R) $= <1, 2, \ldots, n>$ $\quad 1, 2, \ldots, n \in I$
and by matching the relations, under specification, are indexed:

Field (R) $= <R_1, R_2, \ldots R_n>$

The iteration of Field (R) (with parallel matching) yields

$$\begin{aligned}\text{Field Iterate (R)} &= \text{Field}^{\mu-1}\,(\text{Field (R)}) = \\ &= \text{Field}^{\mu-1} <R_1, R_2, \ldots, R_n> = \\ &= \text{Field}^{\mu-2} <<R_{11}, R_{12}, \ldots>, \\ &\quad <R_{21}, R_{22}, \ldots>, \ldots, <R_{n1}, R_{n2}, \ldots>> \\ &= \ldots\end{aligned}$$

Field Iterate (R) is terminated after μ steps, either when all domains arrived at are simple or when R is re-entered as a domain of one of its subdomains. In the latter case the source is asked to provide an entry to the cycle by unzipping at least one of the members of the cycle in an alternative way.

At each step of Field Iterate, matching of topics T_1 and relations R_1 occurs after the realisation of distinction or predication by the source. There can be no relational operator which, applied to relations performs a distinction (or predication).

Conversely, the relational operators used, simulate-after-the-event, the distinction drawn by the source. This reflects the fact that distinction (or predication) is the operation of the source, par excellence.

To image unzipping and matching it is, in general, necessary to employ

1.1. *Permutation:* of coordinates i.e. a rearrangement of the coordinate set of a relation or a product, For example

$$Permutation_{4312} < R_1, R_2, R_3, R_4 > = < R_4, R_3, R_1, R_2 >$$

1.2. *Projection* of a relation to a subset of its coordinates. For example, given an R for which

$$\underline{Field}\ (R) = < R_1, R_2, \ldots, R_p, \ldots, R_n >$$

$\underline{Proj}_p\ (R) = R_p$, R_p can be a relation for example giving $\underline{Field}\ (R) = < R_{p1}, R_{p2} >$ or a set $S_p = \{ S_{pj} \}$, $j \in J$ value set of S_p.

The operation of Field Iterate (R) leads to the specification of the simple domains S in the product $C = (S_1 \times S_2 \times \ldots \times S_n)$ of which R is defined. The specification is further constrained by whole coordinate identity between two different relations R_1 and R_2, if elicited by the source, and imaged as

$$\underline{Proj}_K(\underline{Field}^{\mu}\ (R_1)) = \underline{Proj}_L\ (\underline{Field}^{\nu}(R_2))$$

As a result of this operation, members of the index set are freed to be used for matching with other topics at each step of Field Iterate (if needed). Establishing coordinate identities is a first step in specifying a relation, R. The next step (specification of R as a particular subset in the Cartesian product in which it is defined) is based on identity of some of the values of a coordinate in a relation R_1, and some of the values of the same coordinate in a relation R_2 (R_1, R_2, such that $\underline{Field}\ (R) = < R_1, R_2 >$). For this purpose we use a set of relational operators, including permutation and projection (already specified) and others, chiefly due to Codd. In general, R is imaged as derived from two (or more) relations R_1 and R_2 by the application of an appropriately ordered sequence (Relop) of relational operators. Thus $R = \underline{Relop}\ (R_1, R_2)$. The set of Relational Operators Rel op set includes Cartesian product (over simple or non-simple domains), permutation and projection (applied to one relation).

Given two or more relations specified in extension

R_1 such that $\underline{Field}\ (R_1) = < A, B, \ldots, E, F >$

R_2 such that $\underline{Field}\ (R_2) = < F, G, \ldots, K, L >$

R_3 such that $\underline{Field}\ (R_3) = < L, M, \ldots, P, Q >$[29]

1.3. *Natural Join* $(R_1, R_2) = \{ < r_A, r_B, \ldots, r_E, r_F, r_G, \ldots, r_K, r_L > \text{such that}$

$(r_A \in A) \wedge$
$(r_B \in B) \wedge \ldots \wedge$
$(r_E \in E) \wedge$
$(r_F \in F) \wedge$
$(r_G \in G) \wedge \ldots \wedge$
$(r_K \in K) \wedge$
$(r_L \in L) \wedge$
$(< r_A, r_B, \ldots r_E, r_F > \in R_1) \wedge$
$(< r_F, r_G, \ldots r_K, r_L > \in R_2)$

Natural Join can be applied to more than two relations, for example

$$\text{Natural Join}_{F,L}\ (R_1, R_2, R_3) = \text{Natural Join}_L\ (\text{Natural Join}_F\ (R_1, R_2), R_3)$$

1.4. $$\text{Natural Composition}_F\ (R_1, R_2) = \frac{\underline{Proj}_{A, B, \ldots E, G, \ldots K, L}}{\text{Natural Join}_F\ (R_1, R_2)}$$

1.5. *Restriction* $R_1 \underset{F}{|} \underset{F}{R_2} = \{ < r_A, r_B, \ldots r_E, r_F > \text{such that}$

$(r_A \in A) \wedge$
$(r_B \in B) \wedge \ldots \wedge$
$(r_E \in E) \wedge$
$(r_F \in F) \wedge$
$(< r_F, r_G, \ldots r_K, r_L > \in R_2) \}$

When L = A

[29] The identical coordinates can always appear as exhibited by appropriate permutation. In fact it is possible to work with relationships (in Codd's terms) i.e. equivalence classes of relations under permutation.

1.6. *Cyclic Join*$_{A,F}$ (R_1, R_2) = $\{ < r_A, r_B, \ldots r_E, r_F, \ldots r_K, r_L >$ such that

$(r_A \in A) \wedge$
$(r_B \in B) \wedge \ldots \wedge$
$(r_E \in E) \wedge$
$(r_F \in F) \wedge$
$(r_G \in G) \wedge \ldots \wedge$
$(r_K \in K) \wedge$
$(r_L \in L) \wedge$
$(< r_A, r_B, \ldots r_E, r_F > \in R_1) \wedge$
$(< r_F, r_G, \ldots r_K, r_L > \in R_2) \wedge$
$(r_A = r_L) \}$

1.7. The standard set theoretic operations cannot be applied without certain precautions. Unless the coordinates of the relations are the same, these operations are inapplicable. Even if the coordinates are the same, application may lead to universal and "null" sets (only trivially "relations").

For two relations, defined in the same coordinates

R_3 such that $\underline{\text{Field}}\,(R_3) = < A, \ldots, F >$
R_4 such that $\underline{\text{Field}}\,(R_4) = < A, \ldots, F >$

Intersection $(R_3 R_4)$ = $\{ < r_A, \ldots, r_F >$ such that
$(r_A \in A) \wedge \ldots \wedge (r_F \in F) \wedge$
$(< r_A, \ldots, r_F > \in R) \wedge$
$(< r_A, \ldots, r_F > \in R) \}$

Union (R_3, R_4) = $\{ < r_A, \ldots, r_F >$ such that
$((r_A \in A) \wedge \ldots \wedge (r_F \in F)) \wedge$
$((< r_A, \ldots, r_F > \in R_1) \vee$
$(< r_A, \ldots, r_F > \in R_2) \}$

Complement $_{A \times \ldots \times F}(R_1)$ = $\{ < r_A, \ldots, r_F >$ such that
$(r_A \in A) \wedge \ldots \wedge (r_F \in F)$
$(< r_A, \ldots, r_F > \notin R_1) \}$

2. *Examples*

R_1

A	B	C
I	α	a
I	β	a
2	γ	b
3	δ	c
4	ϵ	d

R_2

C	D
a	u
b	v
b	z
e	x
f	ψ
g	ω

Natural Join$_c$ (R_1, R_2) = R_3

A	B	C	D
I	α	a	u
I	β	a	u
2	γ	b	v
2	γ	b	z

Restriction $(R_1 \mid R_2)_{c\,c}$ = R_4

A	B	C
I	α	a
I	β	a
2	γ	b

Restriction $(R_2 \mid R_1)_{c\,c}$ = R_5

C	D
a	u
b	v
b	z

Composition $_c(R_1, R_2)$ = R_6

A	B	D
I	α	u
I	β	u
2	γ	v
2	γ	z

Complement (Restriction $(R_2 \mid {}^{cc}_{R_2} R_1)) = R_7$

C	D
e	x
f	ψ
g	ω

$Union_{C,D}\ (R_7, R_5) = R_2$

C	D
a	u
b	v
b	z
e	x
f	ψ
g	ω

with the relations

R_8

A	B	C
I	α	a
I	β	a
2	γ	b
3	δ	c
4	ϵ	d
7	ϵ	d

R_9

C	D	A
a	ψ	6
a	ω	5
a	v	I
b	z	2
e	x	3
f	ψ	4
g	ω	4

Cyclic Join$_{C,A}\ (R_8, R_9) = R_{10}$

R_{10}

A	B	C	D	A
I	α	a	v	I
I	β	a	v	I
2	γ	b	z	2

finally for

R_{11}

A	B	C
I	α	a
I	β	a
2	γ	b
3	δ	c
4	ϵ	d

and

R_{12}

A	B	C
3	δ	c
4	ϵ	d
7	ϵ	d

Intersection $_{A,B,C}\ (R_{11}, R_{12})$

A	B	C
3	δ	c
4	ϵ	d

3. *Discussion.* By comparing the tabular forms of the derived and original relations one may easily verify that all relational operators image manipulations of the same sort (a fact inherent in the formal definitions), namely:

identification of one or more shared domains in two different relations.

identification of values of these coordinates shared by the n-tuples of the distinct relations.

a distinction of the subset of a relation as the n-tuples of which those values are ordered members and the subsets of the n-tuples of which these values are not ordered members.

In conclusion, any partition of the cyclic structure (provided it dissects the set of primitives) exhibits two duals that mutually support the specificity of each other. By the same token, the notion of identity of a relation applies either to the whole of the consistent, connected, cyclic relational network R or to any relation R_i in R. It is this feature that enables the interpretation of properties of the relational structure as properties of a Gestalt.

4. *Other Operators.* Many more relational operators can be added to the Rel op set. They are all derivable from appropriate combinations of the operators in the given subset, but are used chiefly for computational economy. For example, Ashby's (1964a) cylinder of a relation, R (such that Field (R) = < A, . . . , K, L, M . . . , Q >) is derived in the following way

$$\text{Cylinder}_L\ (R) = \text{Cartesian product}\ (\underline{\text{Proj}}_{A,\ldots,K,M,\ldots,Q}\ (R) \times (\underline{\text{Proj}}_L\ (R) \cup \text{Complement}_L\ (\underline{\text{Proj}}_L\ (R))))$$

One operator of particular interest is Analogy which relates relations defined in different domains[30] (Pask 1966) by employing *Isomorphism* over the abstract structure imposed upon the n-tuple indexing set $J_\rho = \{ j_\rho \}$ for $\rho = R_1, R_2$ of two relations R_1, R_2 in distinct domains C_1, C_2.

Select two or more coordinates K, L in R_1 and V, T in R_2

i.e. $\{ r_{K_j} : r_{K_j} \in \underline{\text{Proj}}_K\ (R_1) \}$
$\{ r_{L_j} : r_{L_j} \in \underline{\text{Proj}}_L\ (R_1) \}$
$\{ r_{V_j} : r_{V_j} \in \underline{\text{Proj}}_V\ (R_2) \}$
$\{ r_{T_j} : r_{T_j} \in \underline{\text{Proj}}_T\ (R_2) \}$

together with the index sets

$$g_K \in G_K\ ,\ g_L \in G_L\ ,\ g_V \in G_V\ ,\ g_T \in G_T$$

of conditions that specify the values (or n-tuples) r_{K_j} and r_{L_j} of K and L in R_1 and r_{V_j} and r_{T_j} of V and T in R_2. On the condition that mappings N_p, N_q can be drawn over

$$G_K \overset{N_p}{\longleftrightarrow} G_V$$

$$G_L \overset{N_q}{\longleftrightarrow} G_T,$$

non-trivial isomorphisms may be drawn over J_{R_1} and J_{R_2}. These mappings permit R_1 and R_2 to be joined over $(K, J_{R_1R_2}, T)$ and $(L, J_{R_1R_2}, T)$ i.e. realizing two isomorphisms $J_{R_1} \overset{M_1}{\longleftrightarrow} J_{R_2}$ and $J_{R_1} \overset{M_2}{\longleftrightarrow} J_{R_2}$ and finally analogy

---

[30] Analogy is mechanically detectable, as all other operators with the exception of Predication, at the stage of Field Iterate (R), i.e. whenever two different (disjunctive) executions of Field Iterate (R) terminate with two non-identical sets of simple domains.

$$\text{Analogy}_{K,L:V,T}\ (R_1, R_2) = \text{Cyclic Join}_{K,J_{R_1R_2},V:L,J_{R_1R_2},T}\ (R_1, R_2)$$

As cyclic join preserves the identity $i \in J$ of any n-tuple being tied (in Codd's terms), the two mappings M_1, M_2 coalesce into one, M, which leaves us with the condensed graphic notation we employ in practice.

Appendix K. Description and Listing of the EXTEND Program *(B.C.E. Scott and D. Kallikourdis)*

1. *Overall Organisation of the Program.* In the CASTE tutorial mode, a student points to nodes by citing descriptor values. The descriptors are an indexing scheme and each slot in the index is a 'placeholder' for topics and as such may or may not be filled. Thus it is possible for a student to cite a topic which currently does not exist in the subject matter he is learning. For example, a student learning about the motion of falling bodies, might reasonably ask to learn about 'rate of change of accelerations', which does not exist as a topic. When, in a CASTE tutorial, non-existing topics are cited, the student is given the choice of citing some other topic that does exist or of opting to define the new topic and integrate it into the existing subject matter.

If he choses the second option, the student takes the role of source and the EXTEND routine takes over as the control heuristic.

Basically, EXTEND is an adjudicator of new topics; it permits the addition of new topics to the subject matter only if the source successfully shows how the new topic can be derived from other existing topics, in such a way that the total structure retains the properties of cyclicity and consistency. Further, he must specify a task structure for the new topic that encompasses the task structures of subordinates and is encompassed by the task structures of super-ordinates.

The operation of the heuristic is summarised in the flow chart of Diagram K1.

Section 2 of this Appendix presents a worked example, for a part of elementary probability theory. Section 3 is a more technical discussion of the data structure employed and the cyclicity/consistency testing routines.

The program is written in the TELCOMP 2 language. It is shown as an annotated listing (Printout K2 and K3) of Section 4 below. The following is a brief description of the overall organisation of the program.

Part 1 is for entering augmentations to the tutorial heuristic data base. The latter represents entailment relations between

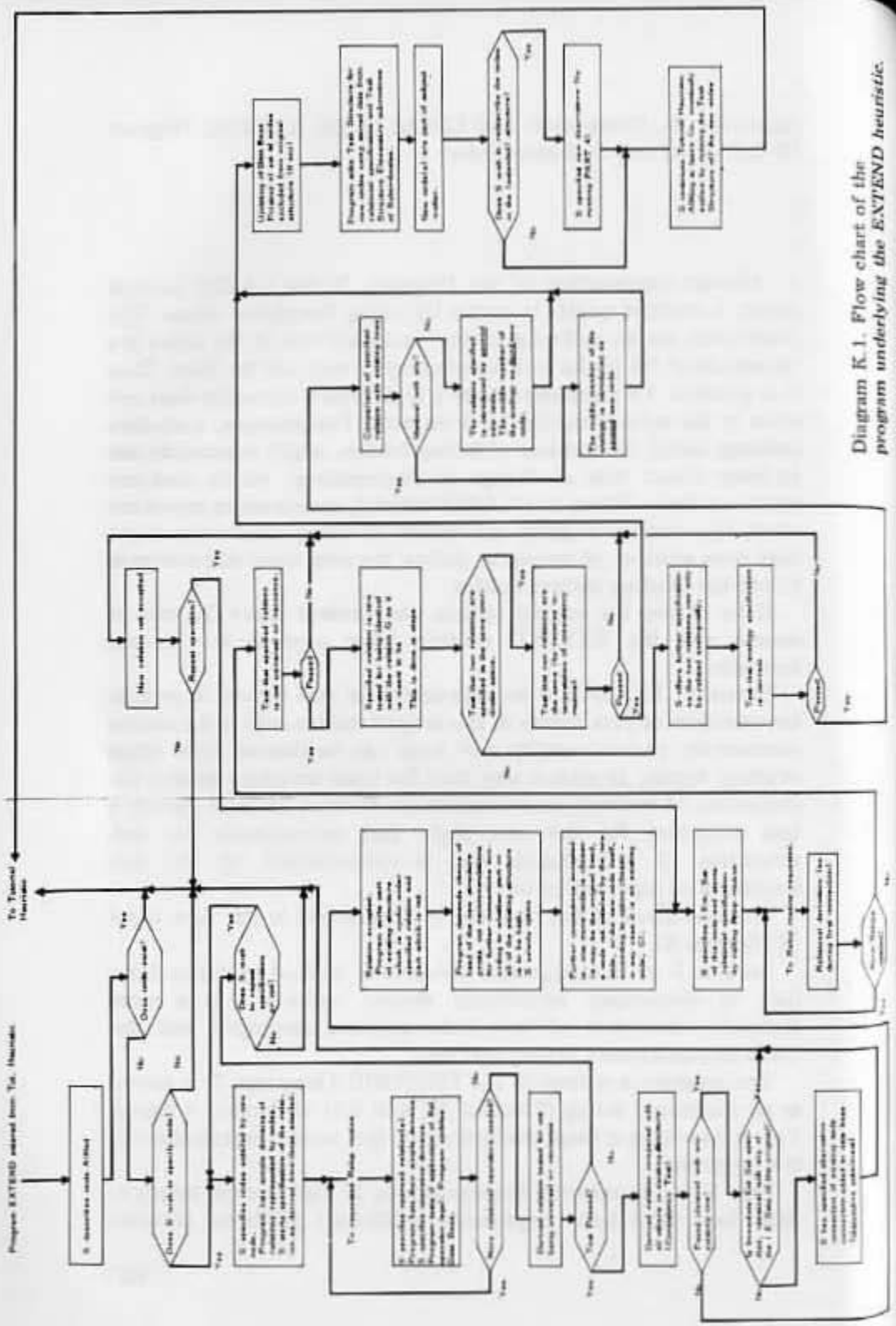


Diagram K.1. *Flow chart of the program underlying the EATEND heuristic.*

nodes; the augmentations permit easy access to any conjunctive substructure within the overall structure.

Part 2 is for entering specifications of how each node (relation) is derived from its subordinates.

Part 3 specifies the form of analogical derivations: notably, coordinate correspondence.

Part 4 accepts descriptor specification and sets up the indexing scheme for node placeholders.

Parts 5,6,7 and 9 are the main control routines as summarised in the flow chart of Diagram K1.

Part 8 tests for non-emptiness, non-universality and consistency of new topics (relations).

Parts 11, 12, 13, 14, 15, 16, 17, and 18 correspond to the application of a relational operator in the derivation of a relation, as follows:

Part No.	Operator
11	Natural join
12	Natural composition
13	Restriction
14	Projection
15	Analogy
16	Complement
17	Union
18	Intersection

Parts 21—44 form the computational heart of the program in testing for cyclicity. The organisation is discussed below (Section 3).

The remaining parts are subsidiary routines, entered from the main routines.

2. *A worked example for the subject matter of elementary probability theory.* The subject matter used in this example is module 1 of elementary probability theory.

Module 1 consists in (a) a set of topics concerning the structure of experiments and the observation of results, (b) a set of topics concerning the abstract modelling of an experiment and (c) a set

of topics stating the form of the analogy that relates the topics of (a) and (b).

Diagram K2 shows the entailment structure. The head nodes are nodes 1, 2, and 3. The primitives are nodes 16–24.

Table 1 lists topic names and the index of the corresponding relation.

Table 2 lists and indexes the simple domains, which represent the coordinate spaces in which each relation is defined.

Table 3 lists, for each relation, the simple domains in the Cartesian product of which the relation is defined.

Table 4 lists the derivation of each relation.

Table 5 lists the descriptors employed and the values they may take.

Table 6 shows, for each node, the value taken by each descriptor. The indexing scheme provides 1895 empty place holders.

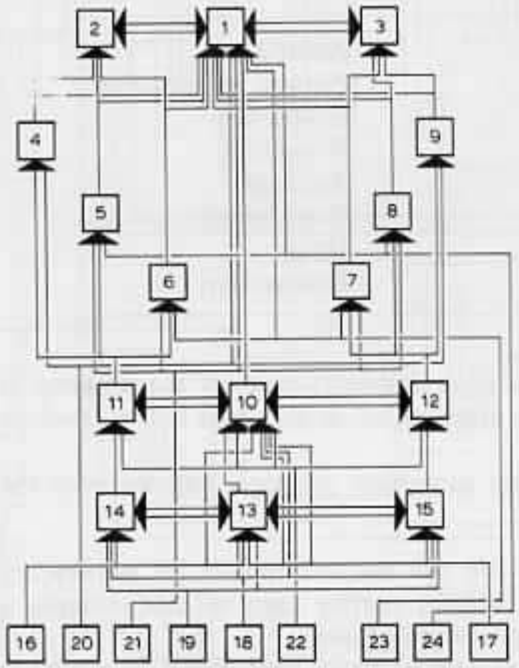


Diagram K.2. Entailment Structure Module 1 of Elementary Probability Theory.

TABLE K.1
List of Topic Names and Indices

Index i	Topic	Relation
1	Theory of Deterministic Experiments	R_1
2	Class of Deterministic Experiments	R_2
3	Structural Model	R_3
4	Complement of Composite Result	R_4
5	Inclusive Composite Results	R_5
6	Exclusive Composite Results	R_6
7	Exclusive Composite Events	R_7
8	Inclusive Composite Events	R_8
9	Complement of Composite Event	R_9
10	Theory of Composite Results	R_{10}
11	Composite Results	R_{11}
12	Composite Event	R_{12}
13	Theory of Simple Results	R_{13}
14	Simple Result	R_{14}
15	Simple Event	R_{15}
16	Result of Observation	R_{16}
17	Event	R_{17}
18	Set and element of Set	R_{18}
19	One-at-once Property	R_{19}
20	Complement of Subset of Set	R_{20}
21	Intersection of Subsets of Set	R_{21}
22	Subset of Set	R_{22}
23	Empty Subset of Set	R_{23}
24	Non-Empty Subset of Set	R_{24}

TABLE K.2
List of Simple Domains

Index name of Simple Domain	Simple Domain
1	Set
2	Element
3	Property
4	Result
5	Event
6	Subset
7	Empty
8	Non-Empty
9	One-at-once
10	Complement
11	Intersection

TABLE K.3
Specification of Coordinate Space in which each Relation is defined (i.e. Field (R) = < Index names of Simple Domains >

Field $(R_1) = <1, 2, 3, 6, 10, 11, 8, 7, 1, 2, 3, 6, 10, 11, 8, 7>$
Field $(R_2) = <1, 2, 3, 6, 10, 11, 8, 7>$
Field $(R_3) = <1, 2, 3, 6, 10, 11, 8, 9>$
Field $(R_4) = <1, 2, 3, 6, 10>$
Field $(R_5) = <1, 2, 3, 6, 11, 8>$
Field $(R_6) = <1, 2, 3, 6, 11, 7>$
Field $(R_7) = <1, 2, 3, 6, 11, 7>$
Field $(R_8) = <1, 2, 3, 6, 11, 8>$
Field $(R_9) = <1, 2, 3, 6, 10>$
Field $(R_{10}) = <1, 2, 3, 6, 1, 2, 3, 6>$
Field $(R_{11}) = <1, 2, 3, 6>$
Field $(R_{12}) = <1, 2, 3, 6>$
Field $(R_{13}) = <1, 2, 3, 1, 2, 3>$
Field $(R_{14}) = <1, 2, 3>$
Field $(R_{15}) = <1, 2, 3>$
Field $(R_{16}) = <3, 4>$
Field $(R_{17}) = <3, 5>$
Field $(R_{18}) = <1, 2, 3>$
Field $(R_{19}) = <3, 9>$
Field $(R_{20}) = <1, 2, 3, 6, 10>$
Field $(R_{21}) = <1, 2, 3, 6, 11>$
Field $(R_{22}) = <1, 2, 3, 6>$

TABLE K.4
Derivation of each Relation

Relational Derivation
$R_{15} = \frac{Res}{3}(R_{18} / \frac{Res}{3}(R_{17} / R_{19}))$
$R_{14} = \frac{Res}{3}(R_{18} / \frac{Res}{3}(R_{16} / R_{19}))$
$R_{13} = \underline{Anal}(R_{14}, R_{15})$ such that: $\frac{Pr}{3,4}(R_{14})$ Isom $\frac{Pr}{3,5}(R_{15})$
$R_{12} = \frac{Nat}{1,2,3}(R_{15}, R_{22})$
$R_{11} = \frac{Nat}{1,2,3}(R_{14}, R_{22})$
$R_{10} = \underline{Anal}(R_{11}, R_{12})$ such that: $\frac{Pr}{3,4}(R_{11})$ Isom $\frac{Pr}{3,5}(R_{12})$
$R_9 = \frac{Nat}{1,2,3,6}(R_{12}, R_{20})$
$R_8 = \frac{Nat}{1,2,3,6}(\frac{Nat}{1,2,3,6}(R_{21}, R_{24}), R_{12})$
$R_7 = \frac{Nat}{1,2,3,6}(\frac{Nat}{1,2,3,6}(R_{21}, R_{23}), R_{12})$
$R_6 = \frac{Nat}{1,2,3,6}(\frac{Nat}{1,2,3,6}(R_{21}, R_{23}), R_{11})$
$R_5 = \frac{Nat}{1,2,3,6}(\frac{Nat}{1,2,3,6}(R_{21}, R_{24}), R_{11})$
$R_4 = \frac{Nat}{1,2,3,6}(R_{20}, R_{11})$
$R_3 = \frac{Nat}{1,2,3,6}(R_7, R_8, R_9)$
$R_2 = \frac{Nat}{1,2,3,6}(R_4, R_5, R_6)$
$R_1 = \underline{Anal}(R_2, R_3)$ such that: $\frac{Pr}{3,4}(R_2)$ Isom $\frac{Pr}{3,5}(R_3)$

The second derivation of relation 13 and the second and third derivations of relations 1 and 10 can be obtained by substitutions of the relations in the second part of the equation.

TABLE K.5
Descriptors and their Values

Descriptor	Values
1 (UNIVERSE)	1 (REAL WORLD) 2 (THEORETICAL) 3 (ABSTRACT WORLD)
2 (ROW)	1st 2nd 3rd 4th 5th 6th 7th
3 (COLUMN)	1st 2nd 3rd 4th 5th 6th 7th 8th 9th
4 (CHARACTER)	1 (STRUCTURED) 2 (COMPLEX) 3 (SIMPLE) 4 (ELEMENTARY) 5 (BASIC)

TABLE K.6
Values of Descriptors for each Topic

Node Index	Descriptor	Value	Node Index	Descriptor	Value
1	1	2	13	1	2
1	2	1	13	2	6
1	3	4	13	3	4
1	4	1	13	4	3
2	1	1	14	1	1
2	2	1	14	2	6
2	3	2	14	3	2
2	4	1	14	4	3

TABLE K.6
Values of Description for each Topic

Node Index	Descriptor	Value	Node Index	Descriptor	Value
3	1	3	15	1	3
3	2	1	15	2	6
3	3	6	15	3	6
3	4	1	15	4	3
4	1	1	16	1	1
4	2	2	16	2	7
4	3	1	16	3	2
4	4	1	16	4	4
5	1	1	17	1	3
5	2	3	17	2	7
5	3	2	17	3	9
5	4	1	17	4	4
6	1	1	18	1	3
6	2	4	18	2	7
6	3	3	18	3	6
6	4	1	18	4	5
7	1	3	19	1	3
7	2	4	19	2	7
7	3	5	19	3	5
7	4	1	19	4	4
8	1	3	20	1	3
8	2	3	20	2	7
8	3	6	20	3	3
8	4	1	20	4	5
9	1	3	21	1	3
9	2	2	21	2	7
9	3	7	21	3	4
9	4	1	21	4	5
10	1	2	22	1	3
10	2	5	22	2	7
10	3	4	22	3	7
10	4	2	22	4	5
11	1	1	23	1	3
11	2	5	23	2	7
11	3	2	23	3	8
11	4	2	23	4	5
12	1	3	24	1	3
12	2	5	24	2	7
12	3	6	24	3	9
12	4	2	24	4	5

An annotated print-out of an extend occasion (Printout K1) follows the set of tables. The student/source cites a new relation conveniently labelled "The intersection of composite events". He specifies how it is derived and goes on to specify how the new relation permits a new way of deriving the superordinate "Exclusive composite events".

3. *Cyclicity and consistency tests of EXTEND*

3.1. *Introduction.* Cyclicity is a property of any conjunctive substructure of a relational network which is consistent and whose head relation is neither void, nor universal in the context of the universe in which it is defined. If these requirements are satisfied, the fragment of knowledge imaged by the conjunctive substructure is knowable.

Second, using a straightforward application of Spencer-Brown's (1969) calculus of indications, we define two conjunctive substructures, representing knowables in respect to the same universe, as equivalent if the forms of the head relations of both are consequential, in the same way, on the form of the universe in which they are defined.

We then extend the argument to handle analogical relations, by restricting the possible forms of the universe to those in which the analogy holds and define, in a similar way, two conjunctive substructures as analogous if the forms of the head relations of both are consequential, in the same way, on the form of the universe in which they are defined.

Having done this, the test procedures of EXTEND can be described meaningfully with the minimum of technicalities.

3.2. *Cyclicity of conjunctive substructures.* A conjunctive substructure is any part of a relational network that when isolated represents one way of constructing the head relation.

Canonically, the structure is represented by a partially ordered set of placeholders ($R_1 \ldots R_n$) and a partially ordered set of relational operators and a mapping that preserves the correspondence between place holders and operator application.

We exhibit the property of cyclicity by noticing that, under the head relation, R_i, we may partition the structure in many ways into pairs of supporting substructures or "duals", R_j, R_k such that $R_i = \underline{\text{Relop}}_1 (R_j, R_k)$ where Relop signifies the application of relational operators to the domains R_j and R_k.

For example, $\underline{\text{Relop}}_1 = \underline{\text{Rest}}\ R_j$ to R_k

This can be stated as: there is a universe (co-ordinate space) where R_i can be reconstructed from R_j and R_k.

We know that R_i, R_j and R_k are distinct in the universe in which R_i is constructed (this is the consistency condition) and we know that R_i is not equal to the universe as a whole nor is it equal to the complement of the universe as a whole (it is not void or universal).

Since the relations in question are closed with respect to any set of relational operations that is employed (by prior specification) it is always possible to write:

$$R_j = \underline{\text{Relop}}_2 (R_i, R_k)$$

and

$$R_k = \underline{\text{Relop}}_3 (R_i, R_j)$$

That is, in such a universe it is also the case that R_j can be constructed from R_i and R_k, and R_j can be constructed from R_i and R_k.

In psychological terms, reconstruction is the property of non-interference; having come to know a head topic all other topics can still be known (remembered). Similarly, in psychological terms, there are cognitive operations in any student's repertoire that are isomorphic to ordered collections of the relational operators which give a closure guarantee.

Such a universe is always derivable from the canonical form of the structure by forming the Cartesian product of the simple domains of the primitives (Prim).

For later reference, this is called the total universe of a conjunctive substructure. The universe obtained by applying Relop, (when the specificity of the simple domains is lost by natural compositions, projections and restrictions) is called the minimal or non-specific universe of the head relation. Diagram K3 shows a simple example.

$< S_1, S_2 >$ and $< S_2, S_3 >$ are the simple domains of R_j, R_k. The total universe is then $S_1 \times S_2 \times S_2 \times S_3$. Application of the relational operator Natural Join to R_2 and R_3 produces R_1. Application of Nat. Join to $<< S_1, S_2 >, < S_2, S_3 >>$ produces $< S_1, S_2, S_3 >$, the minimal universe in which R_1 is constructible.

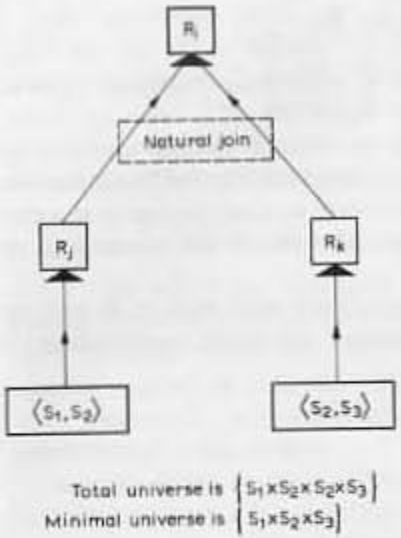


Diagram K.3. Example showing distinction between minimal and total universes.

3.3. *The equivalence of conjunctive substructures.* Conjunctive substructures image ways of coming to know. They are equivalent if what is known is the same in both cases; that not only do both construct the same head relation, but both permit reconstruction of the same relation in all universes[31].

(1) The total universes must be identical.

(2) The minimal universes must be identical.

(3) Whatever construction is made in the total universe, it is always the case that the construction of one head relation is accompanied by the construction of the other head relation.

If these conditions also hold the two substructures image equivalent ways of coming to know the same relation.

[31] In evaluating the equivalence of two procedures (such as computer programs) various criteria are used. For example, one may ask (as we do of knowables), do the procedures compute (bring about) the same relation (this is extensional equivalence). One may ask (as we do not), is the form of computation the same for both procedures (this is intensional equivalence). For knowables, we can distinguish a third sort of equivalence; and ask, with respect to some universe, does executability of one procedure imply executability of the other and vice versa.

Conditions 1 and 2 are determined by simply counting and matching names of primitives and simple domains respectively.

For condition 3, note that for any simple substructure of the form "R_i = Relop (R_j, R_k)" it is possible to write expressions of the form,

(a) $C(R_i) = C(R_j) \wedge C(R_k)$
(b) $C(R_i) = C(R_j) \vee C(R_k)$
(c) $C(R_i) = C(R_j)$

Where C(R) denotes "the construction of R in universe C" and the equals sign means "depends on"; (a) for example, means: "The construction of R_i in universe C depends on the prior construction of R_j and R_k"[32].

Nothing is said about whether or not the R_j, R_k are permitted to be constructed. That is, each expression now represents a permission to construct or distinguish R_i in a universe C, given the constructibility of R_j, R_k.

By beginning from the head node and choosing a path down to the primitives, a string of expressions is obtained, where each expression in the string is contingent for its value on the expressions following.

The string terminates at the primitives. If values are assigned to the primitives (1 = constructed, 0 = not constructed) the expression for the head relation can be evaluated.

The final condition for equivalence of two cognitive structures can now be stated as the requirement that for all possible arrangements of the primitives, whenever one head relation is constructible so is the other.

3.4. *Equivalence of conjunctive substructures, with respect to an analogical universe.* Where the total universes of two substructures are distinct, it may be the case that with respect to some other universe they are not distinct, namely, an analogical universe, which specifies a correspondence between the simple domains of the two universes.

Given that such a correspondence has been defined, equivalence of the structures can be determined by:

[32] The notation is adopted merely for expositary purposes; the expressions are equivalent to expressions in the calculus of indications of Spencer-Brown (1969).

(1) Counting and matching primitives, as before, and treating any corresponding primitives as the same.

(2) Counting and matching simple domains of the minimal universes, as before, and treating any corresponding coordinates as the same.

(3) Comparing constructibility of head relations over just those permutations of values where corresponding primitives always have the same value.

3.5. *Test procedures of EXTEND.* In EXTEND, additions to and modifications of an existing structure are assessed[33]. The questions asked are:

(1) Is the new relation universal or void? This is answered by checking that constructibility of the relation is contingent upon the values of the primitives; i.e. it is never always constructible and never always not constructible.

(2) Is the new relation equivalent to any existing relation? This is the contingency requirement; each distinct relation must have just one place holder. The question is answered by seeing if the procedure for constructing the new relation is equivalent to the procedures that construct other relations.

(3) If the new relation is not a head relation, is the procedure for constructing its superordinates equivalent to existing procedures for constructing its superordinates?

(4) If the new relation is a (new) head relation, is the procedure for constructing the new relation one that necessarily implies the constructibility of all nodes cited as subordinates?

If the latter two questions are answered in the affirmative, the cyclicity requirement is obtained.

The program organisation is complex because it is necessary to cater for many different ways in which an existing structure can be modified. Printout K1 (Section 2) shows the several options available:

(1) to retain the same head node

(2) to instate the new node as head and retain the existing structure

(3) to take the new node as head and delete some of the existing structure

[33] The structure exists because of EXTEND's role in a tutorial. The test routines described are applicable to structures cited de novo.

(4) to take the new node as head and delete all of the existing structure not currently entailed by it.

4. *An annotated listing of the EXTEND program.* The EXTEND listing is given in its TELCOMP form, intended for a terminal user, apart from the elimination of repetitious "type" instructions (i.e. the listing does not repeat "type" for each line of printout). Certain auxiliary instructions, accepted by an external register, are included to avoid asking the user to input step numbers computed by the programme. This register detects the occurrence of any printed out occurrence of the symbol "*" or a pair of "**"; any occurrence of "#" or a string of "#" such as "###" and any occurrence of "@". It is also sensitive to the left arrow "←" output by the computing machine.

The external register has an (adequate) store and it embodies replicas for the Forms numbered in the program.

Let X_i be the ith ($i = 0, 1, \ldots$) *Form* number

Let Y be a *Part* number.

The functioning of the external register is controlled by the symbols reserved (and interpreted by this equipment, not by TELCOMP) as follows:

* means activate register

means receive a first form number (similarly, ## means receive a second form number and so on, for any value of i)

@ means receive a part number placeholder

** means receive data in forms designated

A TELCOMP output is produced (in the program) by enclosing symbols and designations between first and second inverted commas, prefaced by TYPE. For example, consider the statement

TYPE "* # X_0 ## X_1 ### X_2 @@@ Y **"

On execution, the register is actuated, it receives a list of form numbers, for example, $X_0 = 18$, next $X_1 = 19$, next $X_2 = 20$; it receives a part number placeholder and, due to the ** awaits whatever is next output (X_0 in form 18, X_1 in form 19 and X_2 in form 20), and a part number (Y = 7.707, to cite one example from the program). As soon as the external register detects the left

arrow "←" symbol (as a result of a DONE statement, in the program, being executed) it outputs X_0, X_1, X_2 on separate lines (carriage return and line shift between them) and types, on a further line DO PART Y (that is Part 7.707, in this case) thus restarting the internal program.

Such statements are bracketed in the listing: their occurrence clarifies the matter because the external register does those chores which the user would have to do (but need not, logically, do) if the augmenting symbols were not incorporated. The expedient thus disentangles essential, from procedural, human interactions. Moreover, it has some consequence. For the human user cannot easily make sense of EXTEND transactions unless he attends to a display of the existing entailment structure and is freed from most procedural chores. If the display is exhibited on the CASTE processor, as is convenient, always, and mandatory, for student innovation, many other teletypewriter transactions are replaced by the standard CASTE transactions described in the body of the book.

With this preamble, the listing of Printout K3 is about the most intelligible way of stating the capabilities and limitations of EXTEND. Printout K2 (of Part 2.8 and Part 3) indicates how the constructibility conditions, of Section 3.3, are realised in the program.

Printout K1

A segment of user interaction with the EXTEND programme. Certain transactions (those surrounded by large brackets) are suppressed in the operational form: an automatic device (an external register described in Section 4) takes over the chore of typing input part number and restarting program execution. Hence, the segment can be read as though the large bracketed transactions did not occur (in the operational form the user is not bothered with them) and they are included in this printout only to give the reader greater insight into what goes on.

Part 2. Constructibility relations associated with each node, expressed as boolean functions of the corresponding constructibility relations associated with each node's immediate (non analogical) subordinates. Boolean computation is realised by MIN (Minimum of a list) for the operation $\wedge$, MAX (Maximum of a list) for the operation V, arithmatic negation for complement. and $\{+1, -1\}$ for values

Part 3. This part equates values of constructibility relations associated with analogous primitives under side members of analogies. Segments of Part 3 corresponding to analogies inside the part of the structure being tested are executed prior to the execution of Part 2.

Printout K3. Listing of the EXTEND program

```
```

```
12.34 $D#=A(4)+$D#=(A(3)+$D#=(A(2)+$D#=(A(1)+$D#=A(0)
12.35 THEN"ERROR!!! EACH SUBSCR IN COMMON.$SPACE OF
      TUPLE OF ORDER OF APPEARANCE IN CORRESPONDS TO THE
      OF EACH OF THE TWO RELATIONSHIP OF THE INDEX NAME
      IS SIMPLE SUBSCRIPT.E. ENTER TWO TWO CORRESPONDING
      ORDER INDICES FOR EACH DIM SUB IN THE COMMON.SUBSCR"
12.311 STOP
12.312 IF PART 12.31+ THE $(4)1/$D#=A(4)
12.313 TO PART 12.319
12.314 TYPE A.A
12.315 DEFAND 12)12.A(1)
12.316 DEFAND 12)12.A(2)
12.317 STATus IF (A$(2)$1)$1,A(1),A(2)(1)A$(2)$12.A(2)
12.318 END
12.319 TO PART 12.28 IF ST=0
12.30  THEN"SPECIFICATION OF COMMON.SUBSCR IS NOT CORRECT
            RE$(4)"
12.34  TO PART 12.11
12.36 $$(A=A.$1=A.$78,A=A.$1=A
12.37 $$(A=$1)=1
12.38 $1=$1=1
12.39   $$P=A( IF $$(1+$$(12$.A(1)
12.40   $$P=A( IF $$(1+$$(12$.A(1)
12.41   $$$(A=1$$(A=+12$=
12.42   TO PART 12.36 IF $$(12$=+1(A)
12.43   TO PART 12.434 IF $$$(A=+0
12.431  $1=A.(""1)=1=A
12.432  TO PART 12.37 IF $(1)($=+1(1.A 0))
12.433  TO PART 12.*$$))
12.434  $1=$1=1
12.44   THEN"$(A=,A(=1(.$$(12$)(,A=)(1)
12.45   $D=A
12.446  TO PART 12.41 IF $(1)($)=1(.A(1)
12.4465 $D)=A.$D=A.$78(,A=A=1
12.4466 $D)=A( IF $(1)=1$(12$)(,A=)(1)
12.4467 $D)=A( IF $(1)=1$(12$)(,A=)(1)
12.4468 $D)=A( IF $(1)=1$(12$)(,A=)(1)
12.4469 $D)=A( IF $(1)=1$(12$)(,A=)(1)
12.4470 TO PART 12.4472 IF $(1)($=+1(A)
12.4475 TO PART 12.4472 IF $$$(A=+0
12.4476 $D=A.=$$(A=+1=P
12.4477 TO PART 12.4479 IF $$(12$=+1(A)
12.4478 TO PART 12.4479 IF $$(12$=+1(A)
12.4479 TO PART 12.4*1
12.4480 $1=$1=1
12.4481 THEN"$(A=,A(=1(.$$(12$)(,A=)(1)
12.4484 $D=A
12.4485 TO PART 12.4489 IF $$(12$=+1(A)
12.441 $A=A(1)$1=$A(1)=A(1)=$1=$A(1)=A(2)(1)
12.442 $1=$1=1(A)=A-0.$$$(
+-----------------------------------------------+
|12.47   THEN"+++++++++"                        |
|12.48   THEN"ERROR!!! EACH SUBSCR IN COMMON.$SP|
|12.49   THEN"ENTER ONLY TWO"                   |
|12.51   END                                    |
+-----------------------------------------------+
12.43 THEN"ENTER A IF YOU DESIRED SPECIFICATION OF
      ' MODE COMPLETED ,ITHEN"ENTER INTER VALUE NAME
      FOR NEXT RELATIONAL OPERATION"
12.49 DEFAND  RELOPIC$=1
12.491 RELSPEC.FUNCION=RELSPEC
12.492 RELLOC.FUNCION=PHC
12.493 RELOCA.FUNCION=PAGE
```

```
12.494 $A=A(FUNCTION IF RELOPIC$=1=0
12.4*  TO PART 9.93  IF RELOPIC$=1=0 AND FUNCTION
12.491 TO PART 7.241 IF RELOPIC$=1=0 AND FUNCTION
12.57 $=$=1
12.58 ####1
12.59 TO PART RELOPIC$1=1$
Part II Execution
12.11 THEN" SELECTRELATION TO BE RESTRICTED
          FROM <E=$E$$ OF (IS IF"
12.13 TYPE $ IS FORM 4
12.14 TO PART 12.14 IF $=1
12.15 THEN" DA FROM UNIQUE JUST SERIES"
12.16 THEN" SPECIFY AT INDEX NAME"
12.17 DEFAND $=1
12.19 THEN" SPECIFY RESTRICTING RELATION FROM RE=$(1=1=1$
          OF $AMC RE"
12.21 DEFAND $=1
12.22 THEN RELATION IS FORM 9
12.23 TYPE INDICES.I) FOR DE=1(1:DE=A(1) FOR I=$$1:$$
12.24 THEN" RELATION TO BE SPECIFIED WILL BE NAMED"
12.25 THEN 4
12.26 THEN"SELECT SIMPLE DOM=A(=1$1 IN DOMAIN OVER RANGE YOU
          =$=1' RESTRICTION,FROM SET OF SHARED SIMPLE DO=1(=1
          OF THE TWO RELATIONS(POSSIBLY ALL SHARED SIMPLE DO
          STATE NO OF RUN= DO=1(=$"
12.28 DEFAND DO=A(1=1
12.29 $$=A(1(=$D=A(=$$(1
12.31 THEN"SPECIFY EACH SUBSCR IN SP(=1',$=A(1 BY
          (=A=1 OF ORDER OF APPEARANCE IN (DOMS(=A=1$ SP=(1
          OF EACH OF THE TWO RELATIONSHIP BY THE INDEX NAME
          OF SIMPLE SUBSCR(=1(.2.1=1$ THE TWO CORRESPONDING
          ORDER INDICES FOR EACH DIM SUB IN THE OPER.$SUB=1"
12.319 STOP
12.316 IF PART 12.31A FOR SD=A(1(=SD=A(=1
12.317 TO PART 12.322
12.318 TYPE A.A
12.319 DEFAND 12)12.$$(1
12.320 DEFAND 12)12.$$(2
12.321 STATus IF (A$(2)$1)$1,A(1),A(2)(1)A$(2)$12.A(2)
12.322 STOP
12.323 TO PART 12.32 IF ST=0
12.324 THEN"SPECIFICATION OF SPE1=1.SD=A(=1 IS NOT CORRECT
          RE=(=1"
12.326 TO PART 12.11
12.33 THEN"SPE.=A(=1$(12$.A(1) FOR SD=A(1(=SD=A(=1
12.34 $=A(1(=$D=A(=$$(1=$$(1
12.341 $D=A(=1$1=$A(1)=A(1)=$1=$A(1)=A(2)(1)
12.342 $1=$1=1(A)=A-0.$$$(
+-----------------------------------------------+
|12.35   THEN"+++++++++"                        |
|12.37   THEN"ERROR!!! EACH SUBSCR IN COMMON.$SP|
|12.39   THEN"ENTER ONLY TWO"                   |
|12.34   END                                    |
+-----------------------------------------------+
12.41 THEN" ENTER A IF YOU DESIRED SPECIFICATION OF
          MODE COMPLETED ,ITHEN"ENTER INTER VALUE NAME
          FOR NEXT RELATIONAL OPERATION"
12.44 DEFAND  RELOPIC$=1
12.441 RELSPEC.FUNCION=RELSPEC
12.442 RELLOC.FUNCION=PHC
12.443 RELOCA.FUNCION=PAGE
12.444 $A=A(FUNCTION IF RELOPIC$=1=0
12.45 TO PART 9.93 IF RELOPIC$=1=0 AND FUNCTION
12.451 TO PART 7.241 IF RELOPIC$=1=0 AND FUNCTION
```



```
12.46 $=$=1
12.47 ####1
12.48 TO PART RELOPIC$1=1$
Part IV Processor
14.11 THEN" SELECT RELATION TO BE RESTRICTED
          FROM MEMBERS OF (IS IF"
14.13 TYPE $ IS FORM 4
14.14 TO PART 14.14 IF $=1
14.15 THEN" DA FROM UNIQUE JUST SERIES"
14.16 THEN" SPECIFY AT INDEX NAME"
14.17 DEFAND $=1
14.18 THEN RE IS FORM 7
14.19 THEN (RELATION.RE) FOR $(1=1(=$D=A(=1#)
14.21 THEN" RELATION TO BE SPECIFIED WILL BE NAMED"
14.22 THEN 4
14.23 THEN" SELECT SIMPLE DOMAIN TUPLE WHICH YOU PROJECT
          RELATE (E OF SUCH DOM=1(=1"
14.25 DEFAND (THEN1=1
14.26  THEN"SPECIFY (DOMAIN OF (R.E) IN APPEARANCE (S IN
          CORRELATED SPACE OF PROJECTED RELATIONSHIP BY THE
          INTER VALUE OF SIMPLE DOM=1(=1"
14.28  DEFAND ********.R) FOR $(1=1(=$D=A(=1
14.31 $D=A(=1(=$D=A(=1
14.312 (=$(1=1(=1(=$D=A(=1(=1(=1
```

```
15.650  SELECT.FUNCTION*5
15.655  SELECT.FUNCTION*6
15.654  DO 60 I=1,NDIM
15.64 70 PUNT 9.93 IF FUNCTION
15.69 70 PUNT 7.941 IF FUNCTION
15.91 TYPE*,'YOU SHOULD NOT TAKE THE UNION OF THESE RELATIONS'
      1'SEVERAL SELECTIONAL SPECIFICATIONS/OR DELETE'
15.99 PAUSE 2 !! FROM 2
15.971 TYPE*,'ENTER 1 FOR SELECT, 2 FOR REMOVE' +
15.955 IF(ANS) THEN
15.955 70 PUNT 94 IF (SAME)
15.954 70 PUNT  8.22 IF (ANS(1:1) .NE.FUNCTION
15.955 70 PUNT 7.11 IF (ANS(1:1) .NE.FUNCTION
15.99 TYPE '*YOU ARE TRYING ANALYSIS WITHIN THESE RELATIONS'
15.94 70 PUNT 15.91
Part 11 Component
15.11 TYPE*,'SELECT RELATION ?( OR 'DEFAULT')'
      1'FOR NUMBERS OF ITEMS IT+
16.17 PUNT 9 !! FROM 9
16.14 70 PUNT 16.11 IF 941
16.15 PAUSE'ENTER FOUR DIGITS JUST CHECKED'
16.14 TYPE*,'ENTER 82 IN SPECIFIED NAME'
16.17 IDENT 82
16.17 TYPE 82 !! FROM 7
16.19 TYPE '(A$(I),K)' FOR $(A11(I)2=44(8))
16.21 PAUSE'ATTACH TO BE SPECIFIED WILL BE HARDED'
16.22 TYPE 4
16.25 PRINT * THE LOWER/UPPER OF *
16.24 TYPE 82 IN FORM 4
16.29 TYPE*,'ALL BE TAKEN IN COMMON, $PACE IS 81HP.ZOM.3P'
16.26 TYPE 82 IN FORM 4
16.27 $D4=0(1)=$D4=4(8)
16.28 IDENT.41=1400100.82 FOR $D4(1)$D4=4(82
16.29 $TB4=0(1)=$TC4=$C8(1)=0.888)
16.31 $D4=0(1)=$D4=4(82
16.32 IN$(I$$..1)=1$(100,82) FOR $D4(1)$D4=4$C2
16.34  '1111'=4444444'
16.35  1110 1$$PN$(1)=1=1 IN FORM 4
16.37  '1111'=1=1P'
16.41  1$$4
16.39 PAUSE'ENTER 2 !! YOU SHOULD BE SPECIFICATION
      1N NODE COMPLETE .1TH$4=1$( ENTER INDEX
      1FOR NEW' RELATIONAL INFORMATION'
16.42 IF(ANS) RELSPEC(1)
16.421 RELSPEC.FUNCTION=1=RELSPEC(1
16.422 RELSPEC.FUNCTION=1=2
16.423 $N4=1F=$IN=1=1 !! RELSPEC(1)=2
16.43 70 PUNT 9.82 !! RELSPEC(1)=1 AND FUNCTION
16.431 !! PUNT 7.241 !! RELSPEC(1)=2 AND FUNCTION=1
16.44 1=1=1
16.45 <<<<1
16.44 !! PUNT RELSPEC(1)=1
Part 11 Component
17.11 'TEXT' $1.22' RELATIONS !! !! '/1X40,'1)
      1'1=1 1=1=2 !! 11 !!'
17.17 FUNC 4 !! FORM 4
17.14 70 PUNT 17.11 !! 114
17.15 PAUSE'1$ FROM 1(1)) JUST CHECKED'
17.15 TYPE*,'ENTER 1 OR 2'
17.17 IDENT 41
17.18 TYPE 82 !! FORM 7
```



```
17.18 PAUSE 1$0(1)$0:1) FOR $D4(1)$D4=4(1) FOR 1=1(1,4)2
17.181 $N4=1=1=1 =1$(1)$ $1=4 2$4 IDENT $4(8)(1) T! FOR 1=1(1,4)2
17.182 $1=4(1)
17.198  $2=4(1)
17.199 70 PUNT 17.19
17.201  1=4(1=4$(1=4 TO BE SPECIFIED WILL BE HARDED'
17.213 '1=4(1) =
17.21 $D4=1(1=4$D4=4(8)(1)
17.28 IDENT.41=1400100.41(1) FOR $D4(1)($D4=4(8)(1)
17.23 $D4=1(1=4$D4=4(8)(1)
17.24 IN$(I$$..1)=1($(I$,8(1)  FOR $2=1(1)$D4=4(8)(1)
17.461 $4=1=1($(P4(1)$(1),$(1$P4(1)$(1))
17.443 $TC$P4(1=1=$=1.888)
17.47   1111'=4444444'
17.48   1111 1$$PN$(1=1=1=1=1=1=1=1 IN FORM 4
17.49   1111'=1=1P'
17.51   1$$4
17.52 PAUSE'ENTER 2 !! YOU CONSIDER SPECIFICATION OF
      1NODE COMPLETE .1TH$4=1$( ENTER INDEX NAME
      1FOR NEXT RELATIONAL OPERATION'
17.59 IDENT. RELSPEC=1
17.551  RELSPEC(1)=FUNCTION=1=RELSPEC(1
17.553  RELSPEC.FUNCTION=1=2
17.553  RELSPEC.FUNCTION=1=1
17.554  RELSPEC(1)=FUNCTION=1=2
17.555  $TB4=0(1)=$TC4=$C8(1)=0.888)
17.56 70 PUNT 9.93  IF RELSPEC=1=1=8 AND FUNCTION=8
17.561 70 PUNT 7.241 IF RELSPEC=1=1=8 AND FUNCTION=1
17.563 $4=1=1=1=1
17.59 70 PUNT RELSPEC(1=1=2

!! 20 to !! PAUSE check whether all sets of records characters of two
relations

17.71   70 PUNT  17.19 !! 1$4=4(1)(1=$D4=4(1)
17.721  $2=1=4
$8.729 !$$(1=4=4
17.73  $P2=8
17.75  $2(1=4$2=1=1
17.74  $$2=4$2=1=1
17.79  1$(1)  IF (=$(1)$2,1=1=1)(1)(2.82)
17.76  !$$=8  IF NOT( 1=$(1)(1,4)(1=1)(1)(1,1=1)
17.77  !$$=1=4(1$=1=1$
17.74  70 PUNT 17.14 IF $(1)(1)=1=1(2)
17.72  70 PUNT 17.78 IF !$2(1=1=1
17.81  $1$=8
17.815  !$$(1=1=1
17.82  70 PUNT 17.12 IF $(1)$2=1=1=1)
17.8779  $1$=1=1
17.83799  !$$(1=1=8
17.8391  $1$=1=1
17.8778  $1$=1=1$=1
17.8874  $2(1=1$2=1=1
17.8898  !$$=1  IF (=$$1$,1,=1=1)(1)(1$2.82)
17.8876  !$$=8 !!  NOT( (=$$1$(1,=1=1)(1)(1,1=1)
17.8877 !$$(1=1=1$$1=1=1$
17.8878 70 PUNT  17.8874 IF $(1)(1=1=1=1)
17.8879 70 PUNT 17.96 IF !$$(1=1=1
17.8781  $2(1=1=1
17.8782  70 PUNT 17.8873 IF $(1)(1=1=1=1=1)
17.83831  !$$=1=1
17.83883  1) PUNT 7.12= IF $P4=1(1=1)
17.83883  70 PUNT 18.281 IF $P4=1(1=1)
17.8383  70 PUNT 17.281
```

```
17.84  $2=1=1
17.85  70 PUNT 17.98 IF $P4=1(1=1)
17.851   !$$=1=8
17.883  70 PUNT 7.184 IF $P4=1(1=1)
17.94 TYPE*,'YOU SHOULD NOT TAKE THE UNION OF THESE RELATIONS'
17.97 70 PUNT 17.99881
17.99 TYPE*,'THESE RELATIONS SHOULD NOT BE INTERSECTED'
      1'REMOVE OPERATION'
17.99882  TYPE> SPECIFY REL OPERATOR BY INDEX NAME'
17.99883 $D4=1(1  RELSPEC(1)
17.99884 70 PUNT RELSPEC1=1=8
Part 12 Comm
18.11 TYPE*,'SELECT RELATIONS YOU WISH TO TAKE UNION OF'
      1'FOUR NUMBERS OF ITEMS IT+
18.13 TYPE 4 !! FORM 4
18.14 70 PUNT 18.11 IF 841
18.19 TYPE*,'IN THIS ORDER JUST CHECKED'
18.16 TYPE*,'ENTER BY INDEX NAME'
18.17 IDENT 41,842
18.19 TYPE 81, 842  IN FORM 8
18.19 TYPE 1(A$(I$$..1)) FOR $D=1(1)$D4=4(1) FOR 1=841,842
18.191 $4=1(1=1=1 =1$(1)$ $1=4 2$4 IDENT $4(8)(1) T! FOR UNION
18.192 $1(81)
18.193  $2=8(2)
18.199 70 PUNT 17.78
18.111  1111'=RELATIONS TO BE SPECIFIED WILL BE HARDED'
18.193 TYPE 4
18.21 $D4=0(1)=$D4=4(8)(1)
18.28 IDENT.41=1400100.842) FOR $D4(1)($D4=4(8)(1)
18.23 $D4=0(1)=$D4=4(8)(1)
18.24 IN$(I$$..1)=1($(I$,8(1)  FOR $D=1(1)$D4=4(8)(1)
18.441 $4=1=1($(P4(1)$(1),$(1$P4(1)$(1))
18.442 $TC$P4(1=1=$=1.888)
18.47   1111'=4444444'
18.48   1111 1$$PN$(1=1=1=1=1=1=1=1 IN FORM 4
18.49   1111'=1=1P'
18.51   1$$4
18.52 TYPE*,'ENTER 8 !! YOU CONSIDER SPECIFICATION OF
      1NODE COMPLETE .1TH$4=1$( ENTER INDEX NAME
      1FOR NEXT RELATIONAL OPERATOR'
18.55 IDENT.41  RELSPEC(1)
18.551  RELSPEC.FUNCTION=1=RELSPEC(1
18.553  RELSPEC.FUNCTION=1=1
18.555  RELSPEC.FUNCTION=1=2
18.554  IN=1=1(1,IN$$(1=1=1 !! RELSPEC(1=1=8
18.56  70 PUNT 9.93  IF RELSPEC=1=1=8 AND FUNCTION=8
18.57 70 PUNT 7.241 IF RELSPEC=1=1=8 AND FUNCTION=1
18.59 IDENT.41=1=1
18.59 70 PUNT RELSPEC1=1=8
Part 21 (8) on 44 persons of possible relationships in various
constructively produced associated with the previous node, in the
structure. Believe a word by all text and a separated before part 2.
(or before part 3, part 4, 5 for structure being formed complete
example)
21.21  70 PUNT  22 FOR $D4(P4(1)(1)=1=1=1
22.71  70 PUNT  23 FOR $D4(P4(1)(1)=1=1=1
22.71  71 PUNT  45  IF  1$=P4(1=1
22.19  70 PUNT  24 FOR $D4(P4(1)(1)=1=1=1
24.77  77 PUNT  46  IF  1$=P4(1=1
24.22  70 PUNT  25 FOR $D4(P4(1)(1)=1=1=1
25.21  72 PUNT  46  IF  1$=P4(1=1
25.27  77 PUNT  26 FOR $D4(P4(1)(1)=1=1=1
26.11  72 PUNT  46  IF  1$=P4(1=1
26.29  77 PUNT  27 FOR $D4(P4(1)(1)=1=1=1
```



```
27.91  72 PUNT  46  IF  1$=P4(1=1
27.22  22 PUNT  28 FOR $D4(P4(1)(1)=1=1=1
28.81  72 PUNT  46  IF  $1=P4(1=1
28.82  28 PUNT  29 FOR $D4(P4(1)(1)=1=1=1
28.83  72 PUNT  46  IF  1$=P4(1=1
28.87  30 PUNT  30 FOR $D4(P4(1)(1)=1=1=1
28.81  72 PUNT  46  IF  1$=P4(1=1
28.82  30 PUNT  31 FOR $D4(P4(1)(1)=1=1=1
31.91  72 PUNT  45  IF  1$=P4(1=1
31.22  32 PUNT  32 FOR $D4(P4(1)(1)=1=1=1
32.81  72 PUNT  46  IF  1$=P4(1=1
32.85  33 PUNT  33 FOR $D4(P4(1)(1)=1=1=1
33.81  74 PUNT  46  IF  1$=P4(1=1
33.82  34 PUNT  34 FOR $D4(P4(1)(1)=1=1=1
34.81  72 PUNT  46  IF  1$=P4(1=1
34.87  35 PUNT  35 FOR $D4(P4(1)(1)=1=1=1
35.81  74 PUNT  46  IF  1$=P4(1=1
35.82  36 PUNT  36 FOR $D4(P4(1)(1)=1=1=1
36.81  74 PUNT  46  IF  1$=P4(1=1
36.82  37 PUNT  37 FOR $D4(P4(1)(1)=1=1=1
37.81  72 PUNT  46  IF  1$=P4(1=1
37.82  38 PUNT  38 FOR $D4(P4(1)(1)=1=1=1
38.81  72 PUNT  46  IF  1$=P4(1=1
38.82  39 PUNT  39 FOR $D4(P4(1)(1)=1=1=1
39.81  72 PUNT  46  IF  1$=P4(1=1
39.82  40 PUNT  40 FOR $D4(P4(1)(1)=1=1=1
40.81  72 PUNT  46  IF  1$=P4(1=1
40.82  41 PUNT  41 FOR $D4(P4(1)(1)=1=1=1
41.81  72 PUNT  46  IF  1$=P4(1=1
41.82  42 PUNT  42 FOR $D4(P4(1)(1)=1=1=1
42.81  72 PUNT  46  IF  1$=P4(1=1
42.82  43 PUNT  43 FOR $D4(P4(1)(1)=1=1=1
43.81  72 PUNT  46  IF  1$=P4(1=1
43.82  44 PUNT  44 FOR $D4(P4(1)(1)=1=1=1
44.82  45 PUNT  45 FOR $D4(P4(1)(1)=1=1=1
45.881 70 PUNT 49.1 IF $2(1)=1
45.892 70 PUNT 8.771 IF $2(1)=8
45.883 70 PUNT 7.814 IF $2(1)=8
45.884 70 PUNT 19.71 IF $2(1)=4
45.898 70 PUNT 7.71 IF $2(1)=8
45.1 <<<<<1
45.191 DO PUNT 3=1(N$4(1$=4=1=1=1=1=1 FOR $4=1=1(1$$=4=4(1)
45.192 DO PUNT 2
45.193 70 PUNT 8.119
81.11 Part(1)(2) ANALYSIS/1 ENTAIL/MENTS OF SIDE shall MEMBERS
81.12 $4=1(1=1=1=1=1=1=1=1=1=1=1=1 shall THE WIDE NEW OF WHICH
81.13 PAUSE'1=1=1=1=1=1=1=1=1=1=1=
```

```
$1,$0 TO PART $1.$2 IF xxxxxx(x)
$1,$3 xxx
$1,$4 TT PART $1.$2 IF SUB$xx
$1,$5 TO PART $1.x2
$1,$7 $1=$1.+1
$1,$9 xxx($7)=(x+1)&100(x,$)
$1,$8 $1,xxx=0.
$1,$2 TO PART $1.$2
$1,$2 DONE
```

```
$1,$1  (x0.+x0.$x0.$.$P$(xxx)=xyz (x$(x$)(x*1$ IF (x$2
$1,$3  (x)+)
$1,$4  'x$+)
$1,$4  $(PP$(x)  IF SUB('..)+'
$1,$5  $(PP$(x$ IF SUB('..)+')
$1,$$$ $(PP$(x+)/P$(x+)&$P$)
$1,$6  TT PART $1.$3 IF xx'xx$(..)
$1,$7  'x$
$1,$4  TO PART $1.$1 IF $(P$(x+)
$1,$1  $(PP$(x)
$1,$2  TO PART $1.$6 IF )$(P$x+
$1,$2  TO PART $1.$6
$1,$2  $xP(X.)(x)
$1,$2$ $xx$(x)=$; $$ IF xOx( $(P'X$)(x+'X$ IF )$(x
$1,$2  TO PART $1.$2 IF )$(P$x$
$1,$$ DONE
```

```
$1,$0$ @xxxx)(:xNO$X$ $)$+$)='$ IF $$( IF PP$(x')(x$ OF x$)$
$1,$0$ $xx$()$(xx$)()
$1,$0$ )xN($N$(x.)(x$P$)M(')$)=.)$ FOR $x$)($xxx$))
$1,$0x  $$ PART $1.$)
$1,$0$  $P$P$((P$P$)(.)2+)M($2'))$P$() FOR $P$P($x)($)$x$x
$1,$0x $P$xx$()'x $)x$
$1,$0$ DONE
```

```
$1,$1 $$ PART $2.)) FOR )x(.')($P$xP$$ P$)= $$'$ OF $$$ )$$$$
$1,$$ $$ PART $1.$0( FOR )x$.')(')$$' 'P$$$$ )xP$P$$$ P$)= $$'$
$1,$$$ $P$P()x$.$P$P()$x$.$)$x$.)($$)P'x$)$P$$)$.')  $$$ $$'$x')$LY
$1,$$$$ $)$x)x$)$)x$
$1,$$$ $P$P()x$P$P()x)                    )$P$P$x$$ x$) )$ $$T
$1,$$7 $P$P()x$P$P()$x) ))$$'$)$L,$2$'$ $)$.)()$' P$'$
$1,$$$ )$$P$)x( )$ $N$P$)($P$P()$.$)=$N$P$)($P$P()$.')
$1,$$$ )$$P$)x$ )$ $N$P$)($P$P()$.$)=$N$P$)($P$P()$.')
$1,$$) )$$P$)x=)$$)$=)$$P$)
$1,$$2 TO PART $1.$$$ )$ $P$P()x$P$P()$)
$1,$$4  )$$)$=x$
$1,$$$ TO PART $1.$$$ )$ $P$P()x$P$P()$)
$1,$$$ $)$P$)=x'
$1,$$$ $)$P$)=x$
$1,$$$ $)$P$)=x$
$1,$$$ $)$P$)=x$
```